Convergence of HPC and Big Data: Architecture Panel

NITRD Workshop @ Bethesda

Rangan Sukumar, Senior Analytics Architect, Office of the CTO
Cray Inc.
Convergence: Goal and Success

- **Convergence goals** – as a constrained-optimization problem
  - maximize(performance-per-$)
  - minimize($-to-insight)
  - min(operating costs – power, downtime, human_resources)
  - max(architected performance * community productivity) <= budget
  - min(benchmark-performance) >= Scaling_factor
  - max(app-to-app performance variation) <= epsilon

- **Posit**: Real success of convergence is integrating flexibility with heterogeneity
Convergence: Tale of Two Ecosystems

Performance
Languages and tools
Programming models
Access policy
Resource manager and Job-scheduling
Architectural differences

J. Dongarra et al., Exascale computing and Big Data: The next frontier, ACM Communications 2015
Convergence: Tale of Two Ecosystems

- Hypothetical “best” node
  - Sub-Precision
  - Fat nodes (lots of RAM)
  - ASICs (Training + Inference)
  - Persistent node-local storage
  - Run-time code optimization

- Hypothetical “best” node
  - High-Precision
  - Vector extensions
  - CPUs/GPUs with High Bandwidth Memory
  - I/O over memory hierarchy
  - Static code-optimization

J. Dongarra et al., Exascale computing and Big Data: The next frontier, ACM Communications 2015
## Convergence Requirements: Tale of Two Ecosystems

<table>
<thead>
<tr>
<th>Scientific Computing</th>
<th>Enterprise Computing</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Primarily used for</strong></td>
<td><strong>Search/Query, Machine learning</strong></td>
</tr>
<tr>
<td><strong>Philosophy</strong></td>
<td><strong>Send compute to data</strong></td>
</tr>
<tr>
<td><strong>Efficiency via</strong></td>
<td><strong>Distribution</strong></td>
</tr>
<tr>
<td><strong>Scaling expectation</strong></td>
<td><strong>Weak (scale-out)</strong></td>
</tr>
<tr>
<td><strong>Programming model</strong></td>
<td><strong>Map-reduce, SPMD, etc.</strong></td>
</tr>
<tr>
<td><strong>Popular languages</strong></td>
<td><strong>Java, Scala, Python, R</strong></td>
</tr>
<tr>
<td><strong>Design strength</strong></td>
<td><strong>Built-in job fault tolerance over Ethernet</strong></td>
</tr>
<tr>
<td><strong>Access model</strong></td>
<td><strong>Cloud-like</strong></td>
</tr>
<tr>
<td><strong>Preferred algebra</strong></td>
<td><strong>Set-theoretic / Relational</strong></td>
</tr>
<tr>
<td><strong>Memory access</strong></td>
<td><strong>Random</strong></td>
</tr>
<tr>
<td><strong>Storage</strong></td>
<td><strong>Decentralized, Duplication</strong></td>
</tr>
</tbody>
</table>

- **MPI**, **OpenMP**, etc.
- **Map-reduce**, **SPMD**, etc.
## Convergence Requirements: Workflows + Workload

<table>
<thead>
<tr>
<th>Scientific Computing</th>
<th>Enterprise Computing</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data (Structured)</strong></td>
<td>Table, Key-Values, Objects</td>
</tr>
<tr>
<td>Vector, Matrix, Tensor</td>
<td></td>
</tr>
<tr>
<td><strong>Data (Unstructured)</strong></td>
<td>Documents, Images (Camera)</td>
</tr>
<tr>
<td>Mesh, Images (Physics-based)</td>
<td></td>
</tr>
<tr>
<td><strong>Visualization</strong></td>
<td>Word Cloud, Parallel Coordinates, BI Tools</td>
</tr>
<tr>
<td>Voxel, Surface, Point Clouds</td>
<td></td>
</tr>
<tr>
<td><strong>Validation</strong></td>
<td>Manual / Subject matter expert, A/B testing</td>
</tr>
<tr>
<td>Cross-validation (ROC curves, statistical significance)</td>
<td></td>
</tr>
<tr>
<td><strong>Extract, Transform, Load</strong></td>
<td>File-format transformations</td>
</tr>
<tr>
<td>Fourier, Wavelet, Laplace, etc.</td>
<td>e.g. CSV to VRML</td>
</tr>
<tr>
<td>Cartesian, Radial, Toroidal, etc.</td>
<td></td>
</tr>
<tr>
<td><strong>Search (Query)</strong></td>
<td>SQL, SPARQL, etc. (Sum, Average, Group by)</td>
</tr>
<tr>
<td>Properties such as periodicity, self-similarity, anomaly, etc.</td>
<td></td>
</tr>
<tr>
<td><strong>Funding Model</strong></td>
<td>Value-driven (Cost matters)</td>
</tr>
<tr>
<td>Non-profit grand challenge (Answer matters)</td>
<td></td>
</tr>
</tbody>
</table>

## Convergence Requirements: AI Deployment

<table>
<thead>
<tr>
<th>Scientific Computing</th>
<th>Enterprise Computing</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Model</strong></td>
<td>CNN, RNN, LSTM, GAN etc.</td>
</tr>
<tr>
<td><strong>Baseline</strong></td>
<td>Humans, Other ML algorithms</td>
</tr>
<tr>
<td><strong>Parallelism</strong></td>
<td>Data</td>
</tr>
<tr>
<td><strong>Use Case</strong></td>
<td>Speech, Test Image interpretation</td>
</tr>
<tr>
<td><strong>Source File System</strong></td>
<td>Hyper-personalization</td>
</tr>
<tr>
<td><strong>Figure of Merit</strong></td>
<td>Time-to-accuracy, Model-size</td>
</tr>
<tr>
<td><strong>Training Data</strong></td>
<td>O(GBs) per sample, O(10^3) samples, O(10) categories</td>
</tr>
<tr>
<td><strong>Data Model</strong></td>
<td>Relational, Document, Key-Value</td>
</tr>
</tbody>
</table>

- **Domain-specific CNN, RNN, LSTM, GAN etc.**
- **Theoretic e.g. Navier Stokes**
- **Model, Ensemble**
- **Computational Steering Proxy models**
- **Lustre and GPFS**
- **Interpretability, Feasibility**
- **O(GBs) per sample, O(10^3) samples, O(10) categories**
- **HDF5, NETCDF**
- **Time-to-accuracy, Model-size**
- **Speech, Test Image interpretation Hyper-personalization**
- **Data**
- **HDFS, S3, NFS etc.**
- **O(KBs) per sample, O(10^6) samples, O(10^4) categories**
- **Relational, Document, Key-Value**
Convergence: Early Experience @ Cray

Cray CS-Storm 500NX Dense GPU System

Cray CS-Storm 500GT Dense GPU System

Cray XC-50 Accelerated GPU System

Cray UIRKA-GX

CS-STORM 500NX

CS-STORM 500GT

URIRKA-CS

CRAY XC SERIES

CRAY UIRKA-XC

COMPUTE

STORE

ANALYZE
Convergence: Early Experience (Optimism)

Best practices:
- Application fine-tuning / Performance optimization
- High-performance interconnect
- Algorithmic cleverness to trade compute and i/o
- Overlap compute and i/o with programming model

Graph Analytics
- Handle 1000x bigger datasets with a 100x better speed-up with queries

Matrix Methods
- $\begin{bmatrix} \vdots & \vdots \\ \vdots & \vdots \end{bmatrix} \ast \begin{bmatrix} \vdots \\ \vdots \end{bmatrix} \approx \begin{bmatrix} \vdots \end{bmatrix}$
- Get 2-26x over Big Data Frameworks like Hadoop, Spark (for the same cluster-size)

Deep Learning
- 95%+ scalability efficiency that can reduce training time from days to hours
## Convergence: Early Experience (Pessimism)

### ResNet-50 Success

<table>
<thead>
<tr>
<th>Source: NVIDIA</th>
<th>Achieved Images/s</th>
</tr>
</thead>
<tbody>
<tr>
<td>i/o: 9800 images/sec</td>
<td>Compute bound</td>
</tr>
<tr>
<td>i/o: 4800 images/sec</td>
<td></td>
</tr>
</tbody>
</table>

### Rooffine analysis

- **Small Data Region**
- **Power-law Region**
- **Irreducible Error Region**

**Source:** Baidu

### Generalization Error (Log-scale)

<table>
<thead>
<tr>
<th>Training Data Set Size (Log-scale)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Small Data Region</td>
</tr>
<tr>
<td>Power-law Region</td>
</tr>
<tr>
<td>Irreducible Error Region</td>
</tr>
</tbody>
</table>

**Source:** Baidu

### Comparison Table

<table>
<thead>
<tr>
<th>ResNet-50 Success</th>
<th>Time-to-accuracy</th>
<th>How many GPUs?</th>
<th>Scalability Efficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Facebook (Caffe2)</td>
<td>2 days</td>
<td>352 GPUs</td>
<td>90% (large-batch)</td>
</tr>
<tr>
<td></td>
<td>1 hour</td>
<td>256</td>
<td></td>
</tr>
<tr>
<td>IBM PowerAI (Caffe)</td>
<td>50 minutes</td>
<td>256 GPUs</td>
<td>95% (large-batch)</td>
</tr>
<tr>
<td>Google (TensorFlow)</td>
<td>~24 hours</td>
<td>64 TPUs</td>
<td>&gt;90%</td>
</tr>
<tr>
<td>Preferred Networks (Chainer)</td>
<td>15 minutes</td>
<td>1000 GPUs</td>
<td>&gt;90%</td>
</tr>
<tr>
<td>Cray @ CSCS (Tensorflow)</td>
<td>&lt;14 minutes</td>
<td>1000 GPUs</td>
<td>~&gt;95%</td>
</tr>
<tr>
<td>Tencent</td>
<td>&lt; 7 minutes</td>
<td>2048 GPUs</td>
<td>Large batch @ 64K</td>
</tr>
<tr>
<td>Fast.ai on AWS (Cost: $40)</td>
<td>~18 minutes</td>
<td>128 GPUs</td>
<td>Not available (large batch)</td>
</tr>
</tbody>
</table>
Convergence Future: Cray Shasta System

### Vendors

<table>
<thead>
<tr>
<th>Integrated systems</th>
<th>Dell, HPE, Cray, Inspur, NVIDIA...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Provisioning</td>
<td>Bitfusion, Ace, Bright Computing</td>
</tr>
<tr>
<td>Inter-connect</td>
<td>Intel, Cray, Mellanox</td>
</tr>
<tr>
<td>Node architecture</td>
<td>NVIDIA, Facebook, Cray</td>
</tr>
<tr>
<td>Motherboard</td>
<td>Quanta, Supermicron etc.</td>
</tr>
<tr>
<td>xPU</td>
<td>Intel, NVIDIA, AMD, ARM (40+ startups)</td>
</tr>
</tbody>
</table>

### Features

<table>
<thead>
<tr>
<th>Integration, Scaling, Turn-key</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtualization, Scheduling</td>
</tr>
<tr>
<td>OPA, Aries, InfiniBand</td>
</tr>
<tr>
<td>Density, CPU:GPU ratios</td>
</tr>
<tr>
<td>PCIe, NCCL, GPU-Direct</td>
</tr>
<tr>
<td>CPUs, GPUs, ASICs</td>
</tr>
</tbody>
</table>
Convergence Future: Technologies

Convergence is not all hardware.....

Lot more work before convergence can be productive....
Summary: What is in the future?

- **General purpose flexibility**
  - Commodity-like configurations with custom processors, chips

- **Seamless heterogeneity**
  - CPUs, GPUs, FPGAs, ASICs

- **High-performance interconnects for data centers**
  - MPI and TCP/IP collectives, compute on the network

- **Unified software stack with micro-services**
  - Programming environment for performance and productivity

- **Workflow optimization**
  - Match growth in compute, model-size and data with I/O
Thank You
Convergence: What would it take?

**Hardware**

<table>
<thead>
<tr>
<th>System</th>
<th>Function</th>
<th>Ecosystem</th>
</tr>
</thead>
<tbody>
<tr>
<td>Utilization</td>
<td>Application/Codes</td>
<td>Community Productivity</td>
</tr>
<tr>
<td>Peak vs. Sustained, Performance per $</td>
<td>e.g. Deep Learning, Graph analytics</td>
<td>Domain-specific Creativity</td>
</tr>
<tr>
<td>Reliability</td>
<td>Kernel/Motif</td>
<td>Does there an ecosystem of sustainable community (open-source) engagement that enables vertical segments?</td>
</tr>
<tr>
<td>Faults, MTTF, Uptime</td>
<td>e.g. DGEMM, SYRK, ReLU, inner product</td>
<td>Code Portability</td>
</tr>
<tr>
<td>Scalability</td>
<td>Programming Model</td>
<td>Does a user have to rewrite code? Does vendor support code porting for novel architectures?</td>
</tr>
<tr>
<td>Weak and strong</td>
<td>e.g. MR, PGAS, GRPC</td>
<td>Programmability</td>
</tr>
<tr>
<td>System Architecture</td>
<td>Libraries</td>
<td>Does an end-user have to learn a new language or can they launch jobs with modern tools (e.g. notebooks)?</td>
</tr>
<tr>
<td>Interconnect</td>
<td>Collectives</td>
<td>Data Pre-Processing</td>
</tr>
<tr>
<td>e.g. ETH, InfiniBand, Aries</td>
<td>e.g. NCCL, MPI</td>
<td>Does system offer tools to optimize ETL wall-time?</td>
</tr>
<tr>
<td>Provisioning</td>
<td>Data Structure</td>
<td>Data Movement</td>
</tr>
<tr>
<td>e.g. Mesos, Moab, SLURM</td>
<td>e.g. matrix, sequences, unstructured grids</td>
<td>Does system provide ability to run multiple frameworks/applications on the same data?</td>
</tr>
<tr>
<td>Node Architecture</td>
<td></td>
<td></td>
</tr>
<tr>
<td># of xPUs + cache + memory + network</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Disk</td>
<td>Memory</td>
<td>xPU</td>
</tr>
<tr>
<td>Latency</td>
<td>Capacity, Latency</td>
<td>Speed</td>
</tr>
<tr>
<td>i/o</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Software**

**Ecosystem**

**Community Productivity**

- **Domain-specific Creativity**
  - Is there an ecosystem of sustainable community (open-source) engagement that enables vertical segments?

- **Code Portability**
  - Does a user have to rewrite code? Does vendor support code porting for novel architectures?

- **Programmability**
  - Does an end-user have to learn a new language or can they launch jobs with modern tools (e.g. notebooks)?

- **Data Pre-Processing**
  - Does system offer tools to optimize ETL wall-time?

- **Data Movement**
  - Does system provide ability to run multiple frameworks/applications on the same data?
"Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Networking and Information Technology Research and Development Program."

The Networking and Information Technology Research and Development (NITRD) Program

Mailing Address: NCO/NITRD, 2415 Eisenhower Avenue, Alexandria, VA 22314

Physical Address: 490 L'Enfant Plaza SW, Suite 8001, Washington, DC 20024, USA Tel: 202-459-9674, Fax: 202-459-9673, Email: nco@nitrd.gov, Website: https://www.nitrd.gov