High Performance Simulation and Data Analytics

Judy Qiu, Steven Gottlieb

Bingjing Zhang, Peng Bo, Thomas Wiggins, Saliya Ekanayake and Ruizi Li

Intel® PCC at Indiana University

Abstract

The Intel Parallel Computing Center (IPCC) at Indiana University is an interdisciplinary center that aims to address grand challenges in High Performance simulation and data analytics with innovative solutions and software development using Intel architecture. Prof. Judy Qiu's research will focus on novel parallel systems supporting data analytics, while Prof. Steven Gottlieb will focus on adapting the physics simulation code of the MILC Collaboration to the Intel® Xeon Phi™ Processor Family.

MILC on Xeon Phi

IPCC at Indiana University will work on porting the MILC code for Lattice QCD to the Xeon Phi processor.
This will involve:

- Weekly meetings with Intel and NERSC experts as part of a NESAP project involve Arizona, Fermilab and Utah collaborators .
- Ruizi Li, who worked on the MILC port to Knights Corner as part of her Ph.D. thesis, will join Indiana as a postdoc in early November 2015.

Scalable Parallel Interoperable Data Analytics

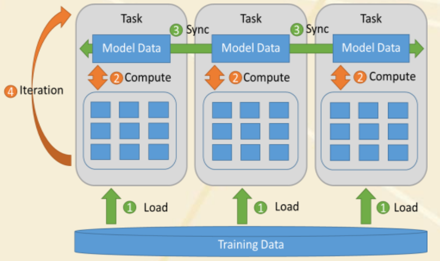
SPIDAL (**S**calable **P**arallel **I**nteroperable **D**ata **A**nalytics **L**ibrary) is a community infrastructure built upon HPC-ABDS concepts like Biomolecular Simulations and Remote Sensing for Polar Science. We have shown that previous standalone enhanced versions of MapReduce can be replaced by **Harp (a Hadoop plug-in)** that offers both data abstractions useful for high performance iteration and communication using best available (MPI) approaches that are portable to HPC and Cloud. This **iterative solver** would enable robustness, scalability, productivity, and sustainability for Data Analytics Link **Mahout**, **MLlib**, **DAAL on HPC-Cloud** and **Deep Learning**.

Harp

Harp is a Hadoop plug-in made to abstract communication by transforming Map-Reduce programming models into Map-Collective models, reducing extraneous iterations and improving performance. It has the following features:

- Hierarchal data abstraction
- Collective communication model
- Pool-based memory management
- BSP-style Computation Parallelism
- Fault tolerance support with checkpointing

Harp-LDA Parallelism

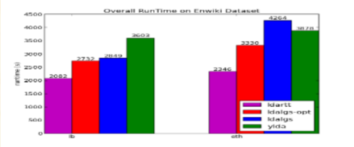


lda-rtt: This Harp-LDA version utilizes model parallelism through rotating global model during one iteration

lda-lgs: This Harp-LDA version synchronizes local-global model once per iteration

Performance Comparison between Harp-LDA and Yahoo! LDA on 200 Iteration

Juliet cluster with Intel Haswell architecture
(100 nodes each with 40 threads and connected with InfiniBand/Ethernet)

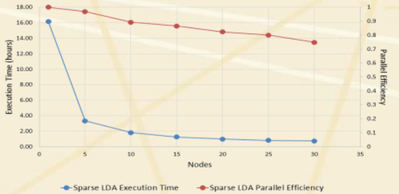


Our experiments use 3,775,554 Wikipedia documents. The training model includes 1 million words and 10K topics. Alpha and beta are both set to 0.01. The number of iterations is 200. Harp-LDA is 42% faster than Yahoo!LDA on the Juliet cluster. Strong scaling tests show Harp-LDA can scale well on Juliet. For the LDA Topic Model:

1. lda-rtt converges quickly. It reaches a higher level of perplexity than what Yahoo!LDA does.
2. lda-rtt performs more model synchronizations in each iteration, which leads to better accuracy and converging speed.

Harp-LDA (lda-rtt) Strong Scaling Test

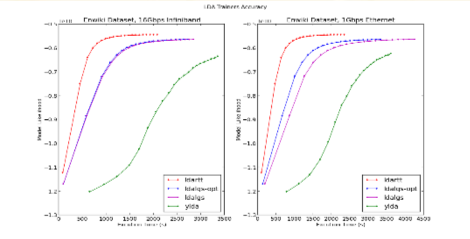
Julietcluster with Intel Haswell architecture
(30 nodes each with 64 threads and connected with InfiniBand)



We run Harp on Latent Dirichlet Allocation (LDA) with Gibbs sampling over a wikipedia dataset where model data is very large. The sparsity of the model is a major challenge, and there is a tradeoff between locking and data replication in shared memory architecture to support concurrent threading. We apply multi-level synchronous and asynchronous optimizations and can achieve better parallel efficiency on the Intel Haswell cluster with 30 nodes (64 threads per node), compared to IU's Big Red II supercomputer with 128 nodes (32 threads per node).

Convergence of LDA Topic Model

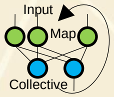
Juliet cluster with Intel Haswell architecture
(100 nodes each with 40 threads and connected with InfiniBand/Ethernet)



Map-Collective

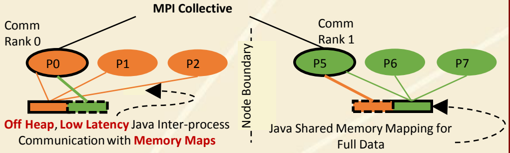
We implement high performance WDA-SMACOF multi-dimensional scaling and WDA-PWC clustering algorithms for the SPIDAL project. These are Map-Collective applications with multiple iterations and substantial communication and computations. Our optimizations include

- **Zero intra-node messaging** – typical MPI would have N processes messaging N-1 others. We reduce it by a factor F where F is processes per node squared
- **Zero GC** – We bring down GC activity to almost zero with reusable Java off heap data structures
- **Minimal Memory** – We statically allocate and reuse all arrays giving minimal memory footprint possible

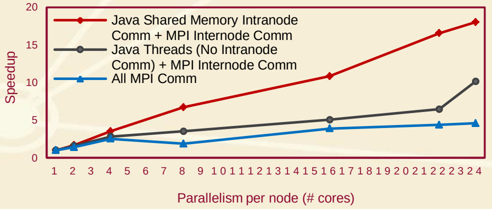


Map-Collective Problem Class

JAVA SHARED MEMORY INTRA-NODE COMM + MPI COLLECTIVES



200K SPEEDUP ON 24 CORE – 48 NODES



We investigate a hybrid use of threads and processes. The performance benchmark runs on a latest **Intel Haswell HPC cluster**. The results show that Intra-node communication poses a considerable overhead if not done through shared memory.

INDIANA UNIVERSITY BLOOMINGTON