MVAPICH
MPI, PGAS and Hybrid MPI+PGAS Library

HiBD
High-Performance Big Data

Power-Performance Modeling of Data Movement Operations on Next-Generation Systems with High-Bandwidth Memory

Talk at ModSim ‘16

by

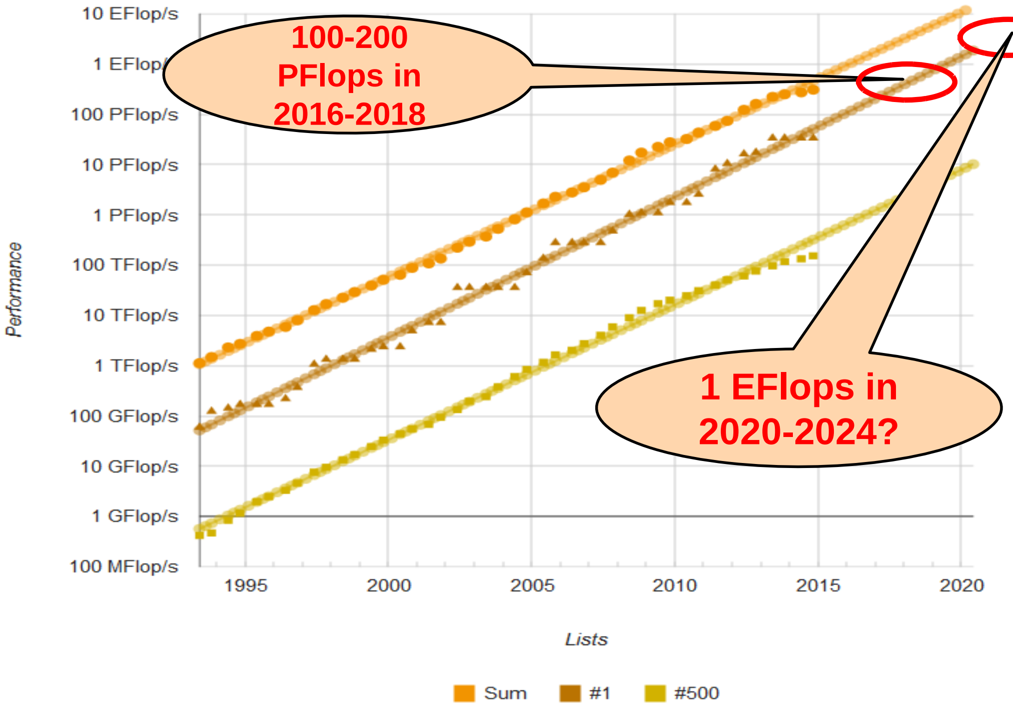
Dhabaleswar K. (DK) Panda

The Ohio State University

E-mail: panda@cse.ohio-state.edu

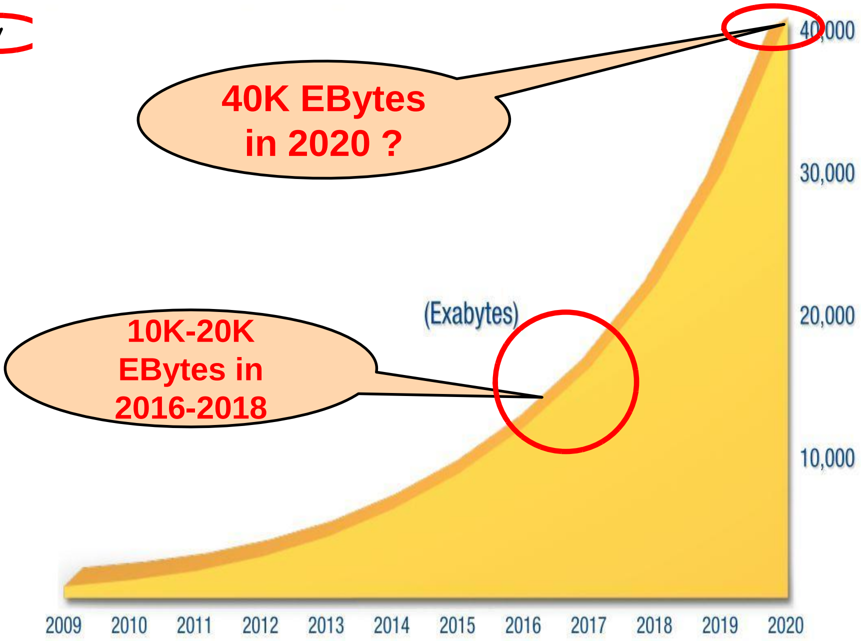
http://www.cse.ohio-state.edu/~panda

High-End Computing (HEC): ExaFlop & ExaByte



100-200 PFlops in 2016-2018

1 EFlops in 2020-2024?



40K EBytes in 2020 ?

10K-20K EBytes in 2016-2018

Source: IDC's Digital Universe Study, sponsored by EMC, December 2012

ExaFlop & HPC

ExaByte & BigData

Drivers of Modern HPC Cluster Architectures

Multi-core Processors

High Performance Interconnects - InfiniBand
<1usec latency, 100Gbps Bandwidth>

Accelerators / Coprocessors
high compute density, high performance/watt
>1 TFlop DP on a chip

SSD, NVMe-SSD, NVRAM

- Multi-core/many-core technologies
- Remote Direct Memory Access (RDMA)-enabled networking (InfiniBand and RoCE)
- HBM, Solid State Drives (SSDs), Non-Volatile Random-Access Memory (NVRAM), NVMe-SSD
- Accelerators (NVIDIA GPGPUs and Intel Xeon Phi)

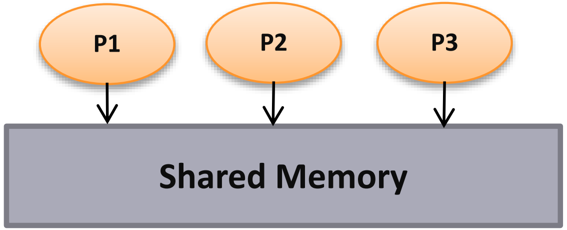
Tianhe – 2

Titan

Stampede

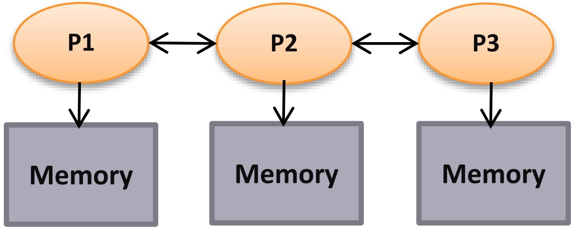
Tianhe – 1A

Parallel Programming Models Overview



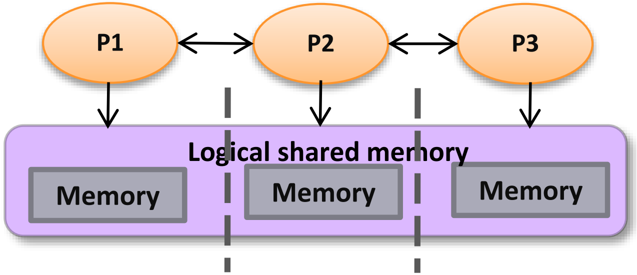
Shared Memory Model

SHMEM, DSM



Distributed Memory Model

MPI (Message Passing Interface)



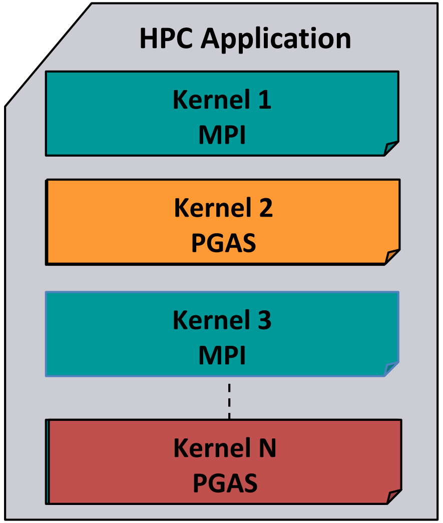
Partitioned Global Address Space (PGAS)

Global Arrays, UPC, Chapel, X10, CAF, …

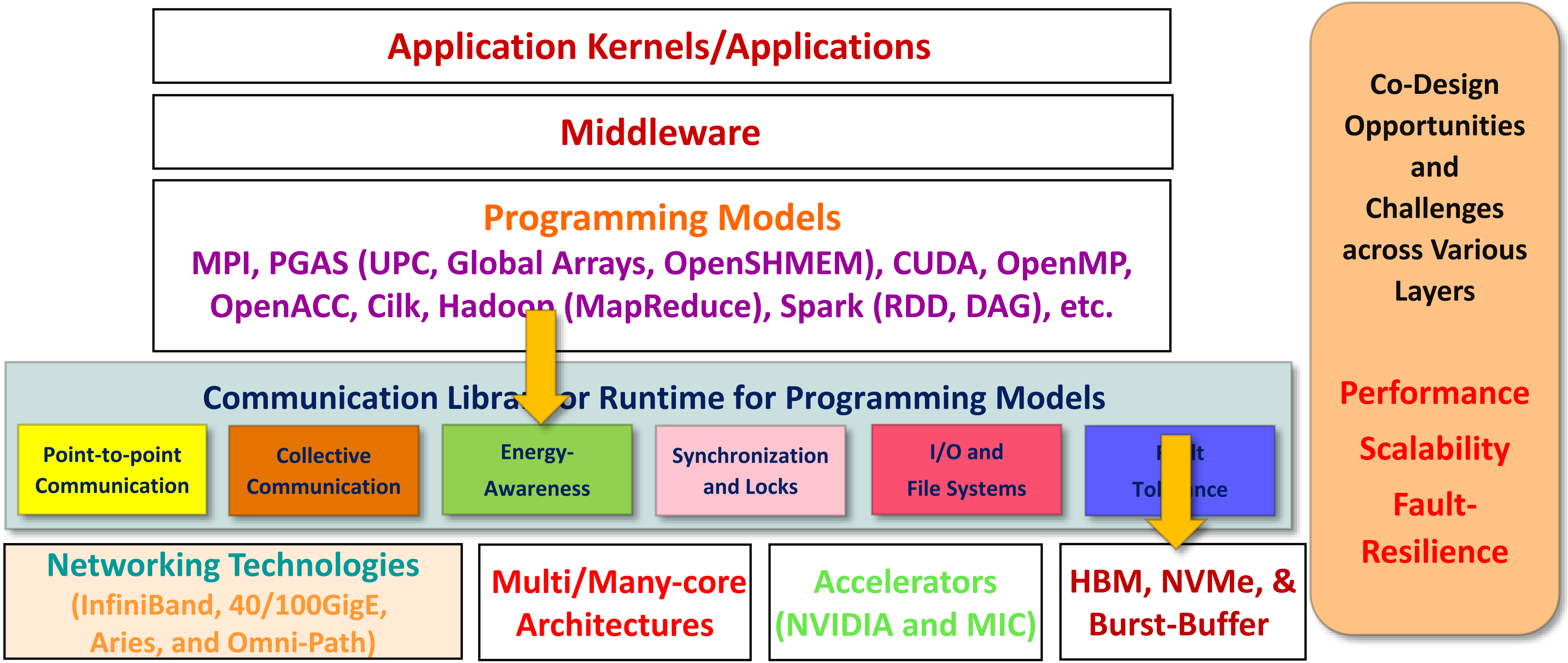
- Programming models provide abstract machine models
- Models can be mapped on different types of systems
  - e.g. Distributed Shared Memory (DSM), MPI within a node, etc.
- PGAS models and Hybrid MPI+PGAS models are gradually receiving importance

Hybrid (MPI+PGAS) Programming

- Application sub-kernels can be re-written in MPI/PGAS based on communication characteristics
- Benefits:
  - Best of Distributed Computing Model
  - Best of Shared Memory Computing Model



Designing Communication Libraries for Multi-Petaflop and Exaflop Systems: Challenges



Application Kernels/Applications

Middleware

Programming Models

MPI, PGAS (UPC, Global Arrays, OpenSHMEM), CUDA, OpenMP, OpenACC, Cilk, Hadoop (MapReduce), Spark (RDD, DAG), etc.

Communication Library or Runtime for Programming Models

Point-to-point Communication

Collective Communication

Energy-Awareness

Synchronization and Locks

I/O and File Systems

Fault Tolerance

Networking Technologies (InfiniBand, 40/100GigE, Aries, and Omni-Path)

Multi/Many-core Architectures

Accelerators (NVIDIA and MIC)

HBM, NVMe, & Burst-Buffer

Co-Design Opportunities and Challenges across Various Layers

Performance

Scalability

Fault-Resilience

Data Movement is not just Network but also requires CPU & Memory

- Data movement operations (Send, Recv, Collective …) requires:
    - CPU: executes instructions to prepare the transfer and polls for the completion of the operation
    - Memory: data resides in memory, hence any data movement requires memory operations to read/write
    - Network: to transfer the data between different domain spaces
- Above components have an impact on the power and energy consumption
- Network components (HCA, Cables and Switches) have the smallest share on the power budget

Several Data Movement Paths

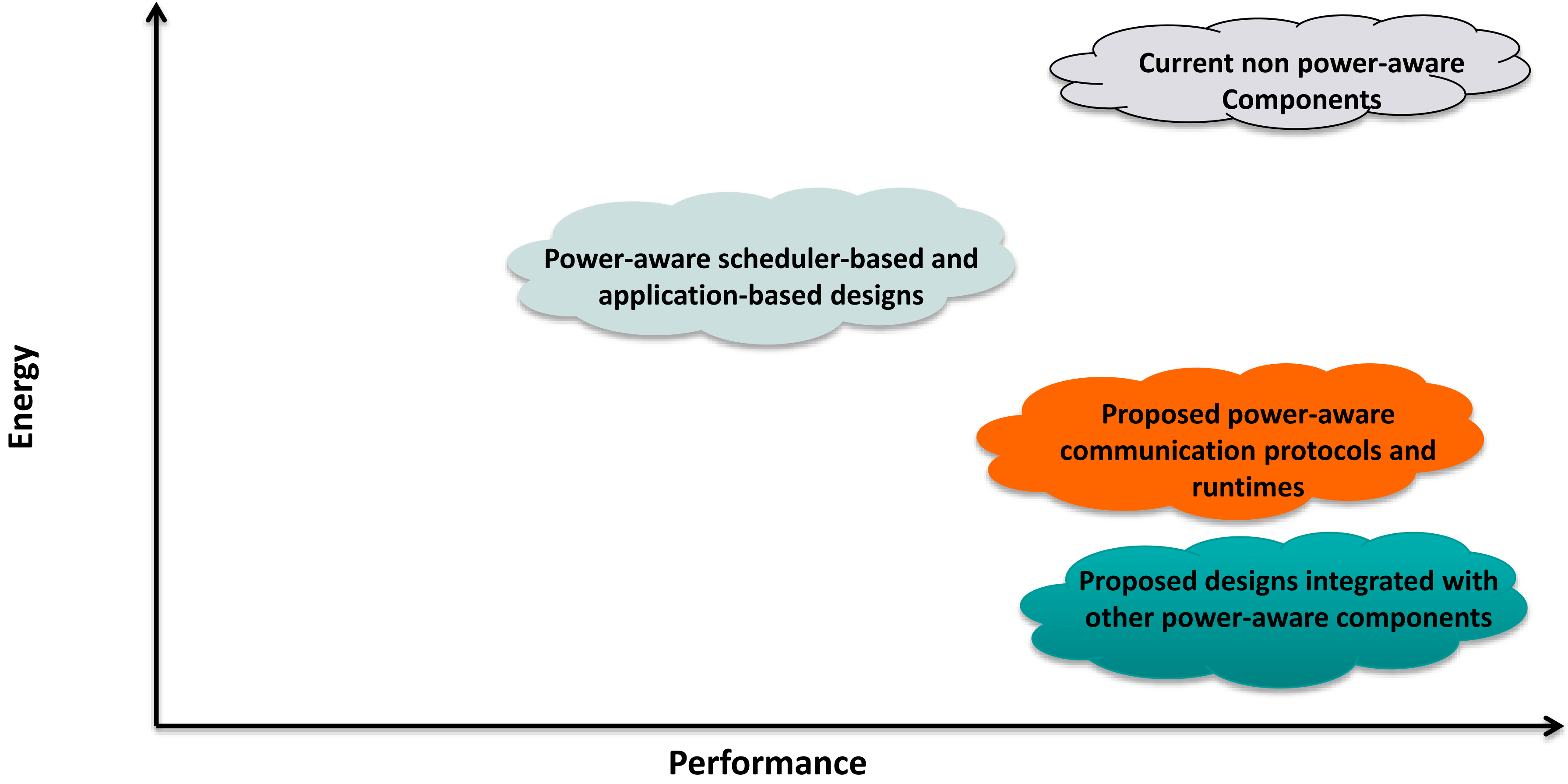
- Multi-level hierarchy and heterogeneity of memory
  - MCDRAM (different configurations)
  - NVMe-based (RDMA capabilities)
  - GPU-based unified memory with CAPI and NVLINK
- Heterogeneity on the processing elements (CPU)
  - Lightweight cores like KNL (SPMD, SIMD)
  - Accelerator cores like GPU (SMPT)
  - Traditional Heavy cores (Xeon, Power)
  - Different memory load/store/barrier costs
- Hierarchical and Heterogeneous Networks
  - NoCs (KNL)
  - NVLINK (GPU)
  - PCIe, QPI, CAPI
  - RDMA (IB, OPA)
  - Different characteristics to access memory

Different Performance and Power Costs to move data from/to High Bandwidth Memory

Current Protocols for MPI+X are Designed for Performance

- MPI+X is seen as the target model for HBM-based systems
- Current MPI+X data movement protocols and runtimes are designed with Performance as the main and only metric
- Using performance-centric models like LogGP to build and design these protocols
- In future power-constrained environments, it is critical to redesign communication protocols and runtimes from the ground-up with energy conservation as one of the core requirements

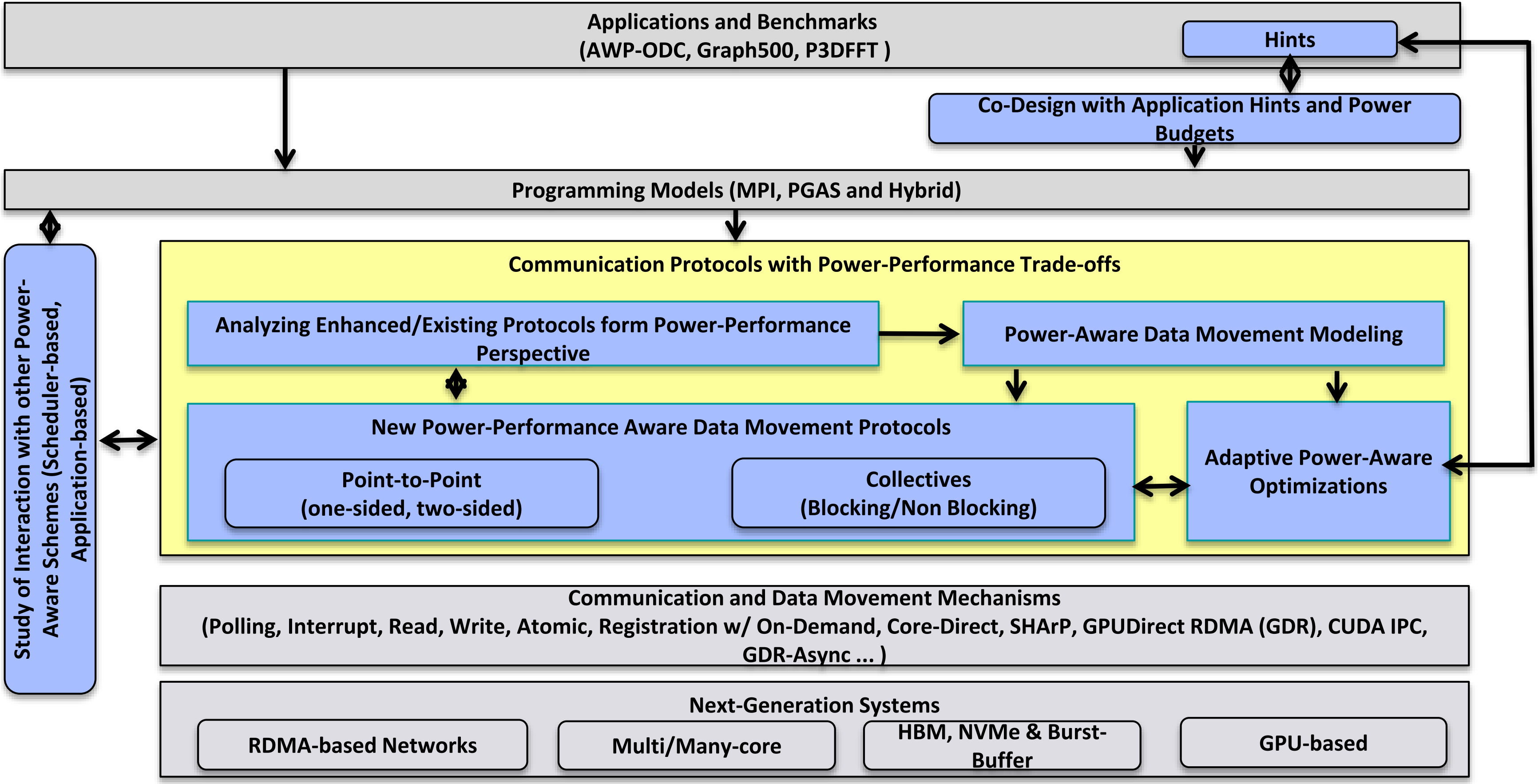
Envisioned Power-Aware Approaches


Current non power-aware Components
Power-aware scheduler-based and application-based designs
Proposed power-aware communication protocols and runtimes
Proposed designs integrated with other power-aware components
Energy
Performance

Objectives and Research Challenges

- Understanding the behavior of current communication protocols with respect to power consumption
- Designing Parallel Communication Models with Power-Performance trade-offs (PCM-PPT) as the first building-block
- Designing a Power-LogGP model (PLogGP) which extends the traditional LogGP model with new parameters to capture the interplay between power and performance
- Design Power-aware runtimes for next generation systems with HBM and accelerators
- Exploiting machine learning algorithms such as regression and classification to better model power-performance complexity

Our Vision



Applications and Benchmarks
(AWP-ODC, Graph500, P3DFFT)

Hints

Co-Design with Application Hints and Power Budgets

Programming Models (MPI, PGAS and Hybrid)

Communication Protocols with Power-Performance Trade-offs

Analyzing Enhanced/Existing Protocols form Power-Performance Perspective

Power-Aware Data Movement Modeling

New Power-Performance Aware Data Movement Protocols

Point-to-Point
(one-sided, two-sided)

Collectives
(Blocking/Non Blocking)

Adaptive Power-Aware Optimizations

Study of Interaction with other Power-Aware Schemes (Scheduler-based, Application-based)

Communication and Data Movement Mechanisms
(Polling, Interrupt, Read, Write, Atomic, Registration w/ On-Demand, Core-Direct, SHArP, GPUDirect RDMA (GDR), CUDA IPC, GDR-Async ...)

Next-Generation Systems

RDMA-based Networks

Multi/Many-core

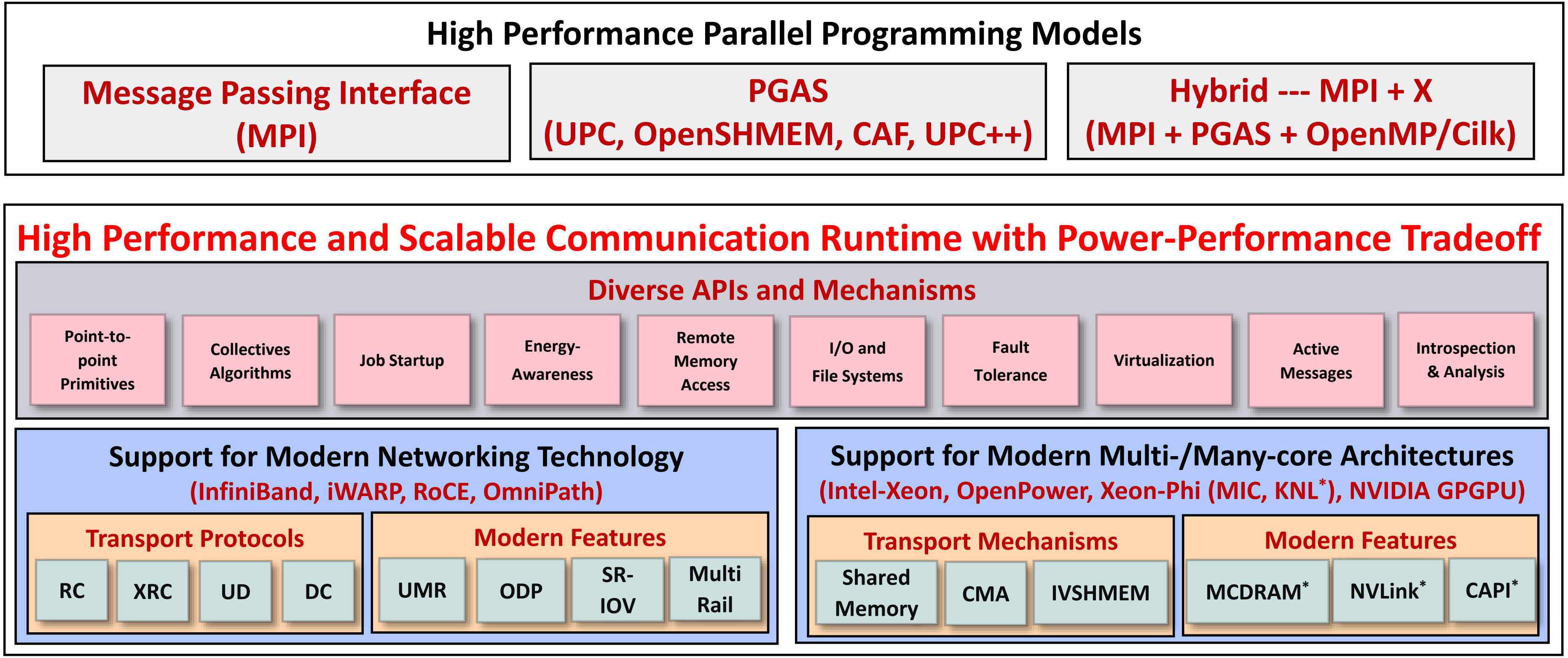
HBM, NVMe & Burst-Buffer

GPU-based

Overview of the MVAPICH2 Project

- High Performance open-source MPI Library for InfiniBand, Omni-Path, Ethernet/iWARP, and RDMA over Converged Ethernet (RoCE)
  - MVAPICH (MPI-1), MVAPICH2 (MPI-2.2 and MPI-3.0), Started in 2001, First version available in 2002
  - MVAPICH2-X (MPI + PGAS), Available since 2011
  - Support for GPGPUs (MVAPICH2-GDR) and MIC (MVAPICH2-MIC), Available since 2014
  - Support for Virtualization (MVAPICH2-Virt), Available since 2015
  - Support for Energy-Awareness (MVAPICH2-EA), Available since 2015
  - Support for InfiniBand Network Analysis and Monitoring (OSU INAM) since 2015
  - **Used by more than 2,625 organizations in 81 countries**
  - **More than 383,000 (> 0.38 million) downloads from the OSU site directly**
  - Empowering many TOP500 clusters (Jun '16 ranking)
    - 12th ranked 519,640-core cluster (Stampede) at TACC
    - 15th ranked 185,344-core cluster (Pleiades) at NASA
    - 31st ranked 76,032-core cluster (Tsubame 2.5) at Tokyo Institute of Technology and many others
  - Available with software stacks of many vendors and Linux Distros (RedHat and SuSE)
  - http://mvapich.cse.ohio-state.edu
- Empowering Top500 systems for over a decade
  - System-X from Virginia Tech (3rd in Nov 2003, 2,200 processors, 12.25 TFlops) ->
  - Stampede at TACC (12th in Jun'16, 462,462 cores, 5.168 Plops)

Architecture of MVAPICH2 Software Family



High Performance Parallel Programming Models

Message Passing Interface (MPI)

PGAS (UPC, OpenSHMEM, CAF, UPC++)

Hybrid --- MPI + X (MPI + PGAS + OpenMP/Cilk)

High Performance and Scalable Communication Runtime with Power-Performance Tradeoff

Diverse APIs and Mechanisms

Point-to-point Primitives

Collectives Algorithms

Job Startup

Energy-Awareness

Remote Memory Access

I/O and File Systems

Fault Tolerance

Virtualization

Active Messages

Introspection & Analysis

Support for Modern Networking Technology (InfiniBand, iWARP, RoCE, OmniPath)

Transport Protocols: RC, XRC, UD, DC

Modern Features: UMR, ODP, SR-IOV, Multi Rail

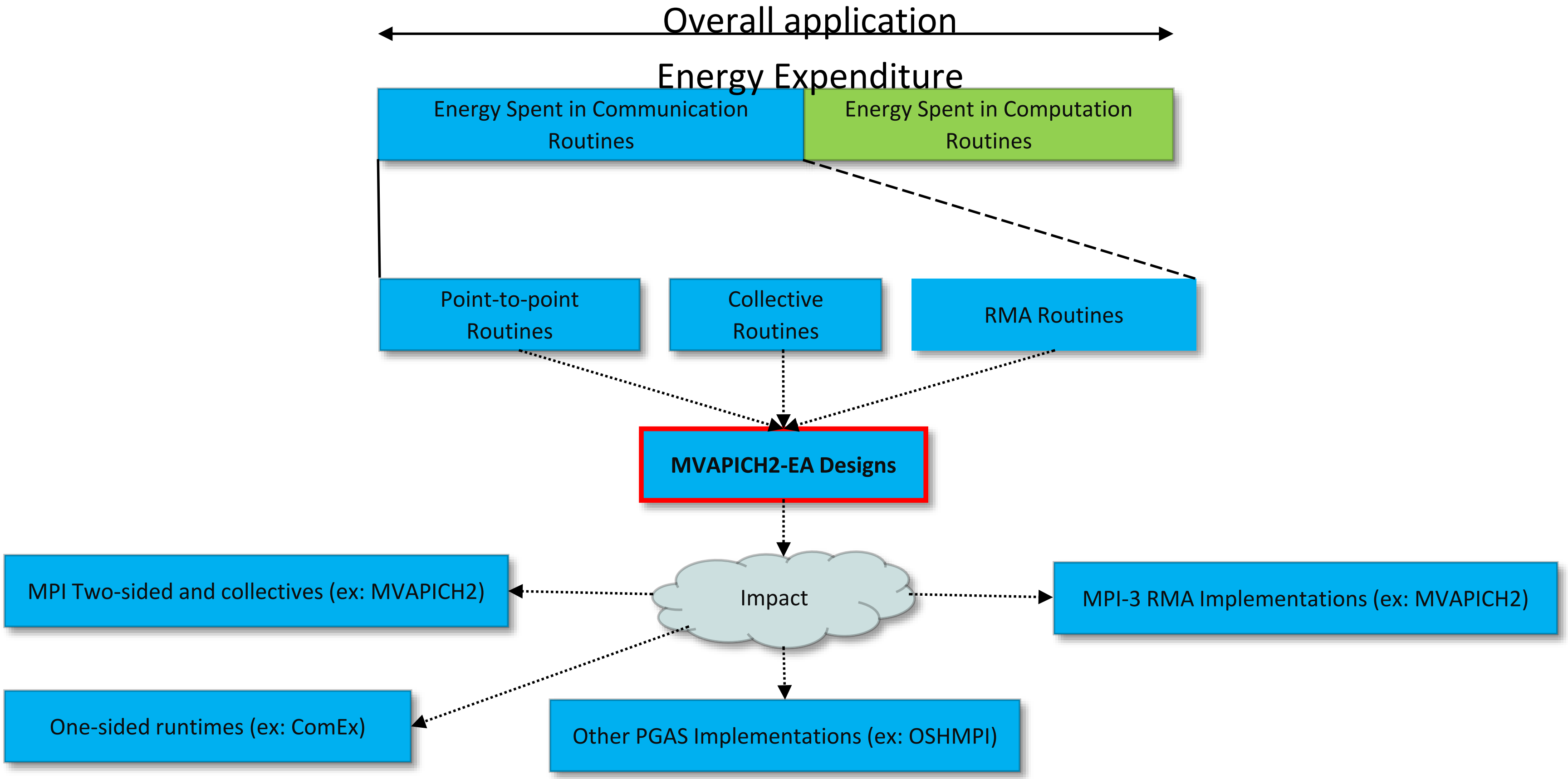
Support for Modern Multi-/Many-core Architectures (Intel-Xeon, OpenPower, Xeon-Phi (MIC, KNL*), NVIDIA GPGPU)

Transport Mechanisms: Shared Memory, CMA, IVSHMEM

Modern Features: MCDRAM*, NVLink*, CAPI*

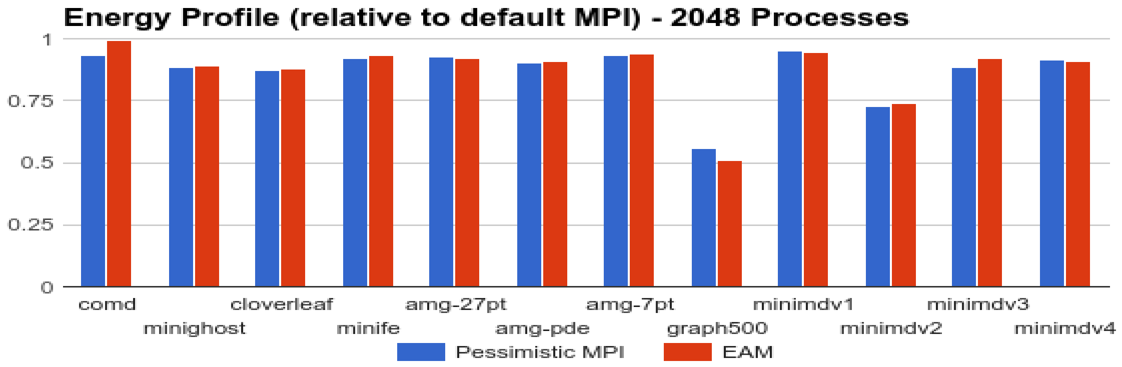
* Upcoming

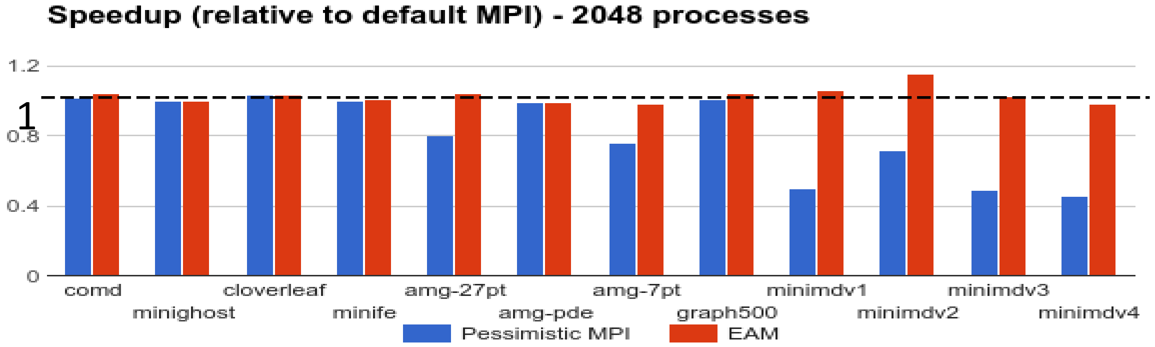
Designing Energy-Aware (EA) MPI Runtime


Overall application
Energy Expenditure
Energy Spent in Communication Routines
Energy Spent in Computation Routines
Point-to-point Routines
Collective Routines
RMA Routines
MVAPICH2-EA Designs
Impact
MPI Two-sided and collectives (ex: MVAPICH2)
MPI-3 RMA Implementations (ex: MVAPICH2)
One-sided runtimes (ex: ComEx)
Other PGAS Implementations (ex: OSHMPI)

MVAPICH2-EA: Application Oblivious Energy-Aware-MPI (EAM)

- An energy efficient runtime that provides energy savings without application knowledge
- Uses automatically and transparently the best energy lever
- Provides guarantees on maximum degradation with 5-41% savings at <= 5% degradation
- Pessimistic MPI applies energy reduction lever to each MPI call



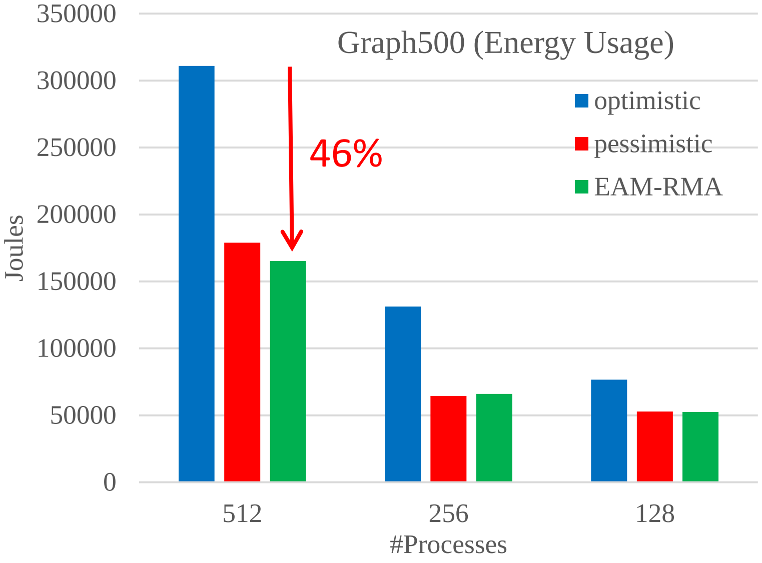


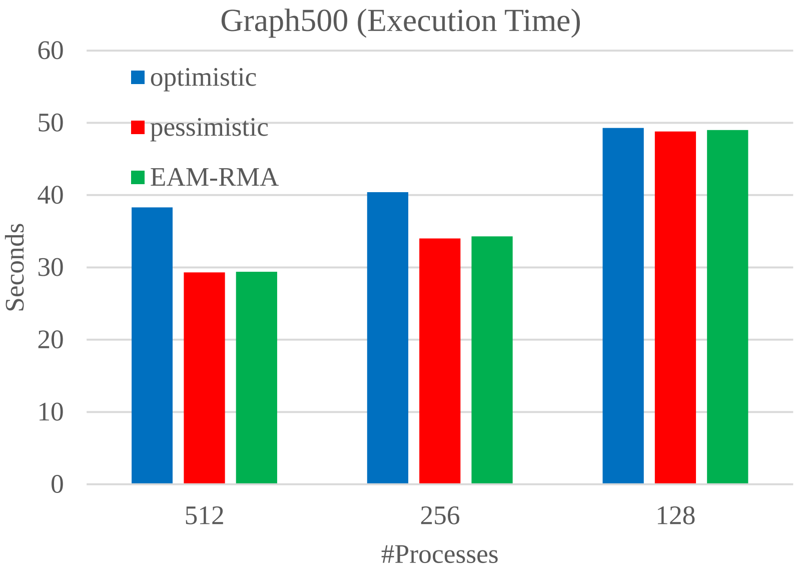
A Case for Application-Oblivious Energy-Efficient MPI Runtime A. Venkatesh, A. Vishnu, K. Hamidouche, N. Tallent, D. K. Panda, D. Kerbyson, and A. Hoise, Supercomputing '15, Nov 2015 *[Best Student Paper Finalist]*

Energy-Aware MVAPICH2 & OSU Energy Management Tool (OEMT)

- MVAPICH2-EA 2.1 (Energy-Aware)
  - A white-box approach
  - New Energy-Efficient communication protocols for pt-pt and collective operations
  - Intelligently apply the appropriate Energy saving techniques
  - Application oblivious energy saving
- OEMT
  - A library utility to measure energy consumption for MPI applications
  - Works with all MPI runtimes
  - PRELOAD option for precompiled applications
  - Does not require ROOT permission:
    - A safe kernel module to read only a subset of MSRs
- Publicly available since August ‘15

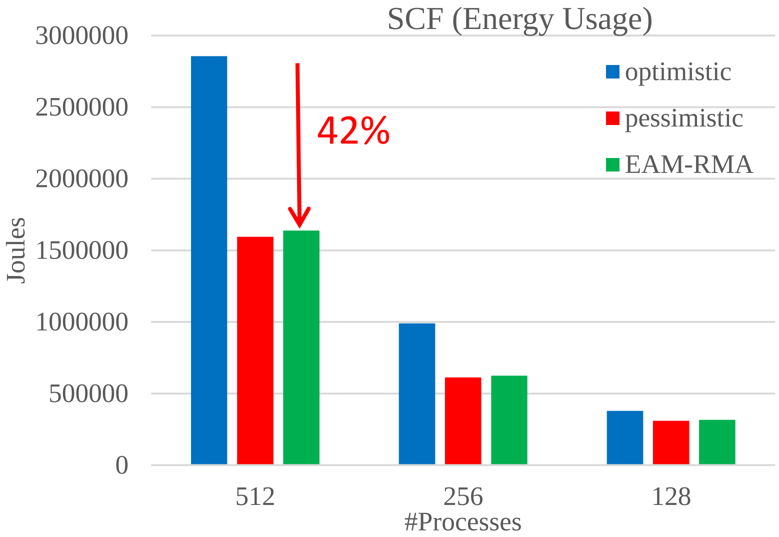
MPI-3 RMA Energy Savings with Proxy-Applications

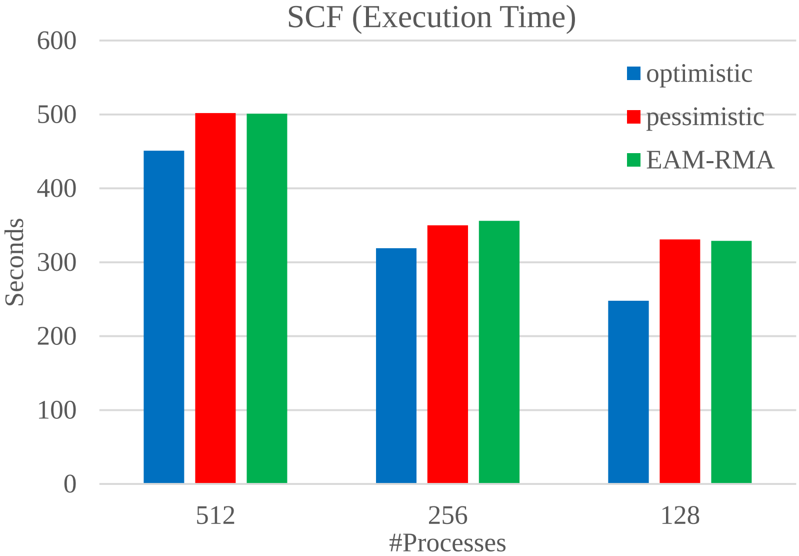




- MPI_Win_fence dominates application execution time in graph500
- Between 128 and 512 processes, EAM-RMA yields between 31% and 46% savings with no degradation in execution time in comparison with the default optimistic MPI runtime

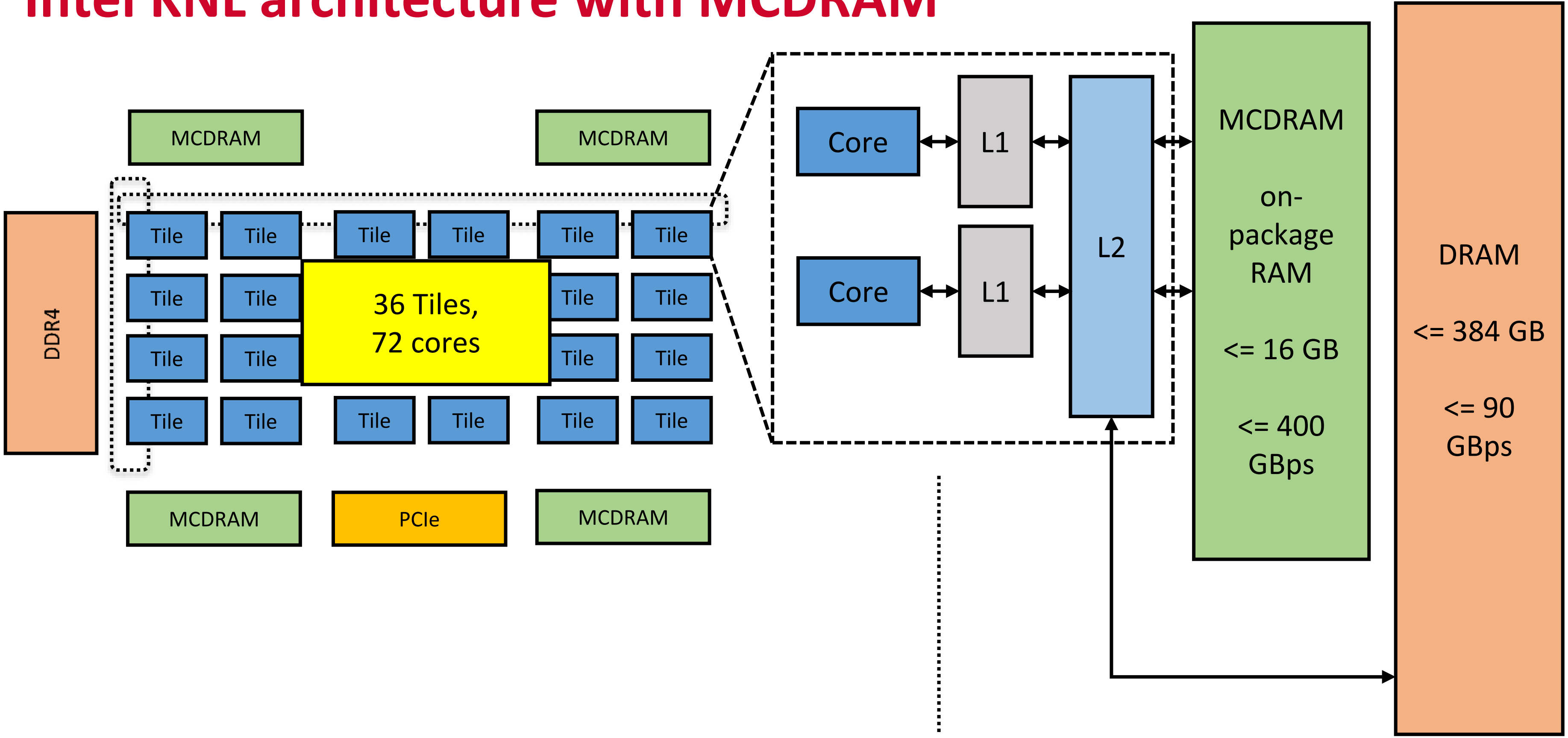
MPI-3 RMA Energy Savings with Proxy-Applications





- SCF (self-consistent field) calculation spends nearly 75% total time in MPI_Win_unlock call
- With 256 and 512 processes, EAM-RMA yields 42% and 36% savings at 11% degradation (close to permitted degradation ρ = 10%)
- 128 processes is an exception due 2-sided and 1-sided interaction
- MPI-3 RMA Energy-efficient support will be available in upcoming MVAPICH2-EA release

Intel KNL architecture with MCDRAM



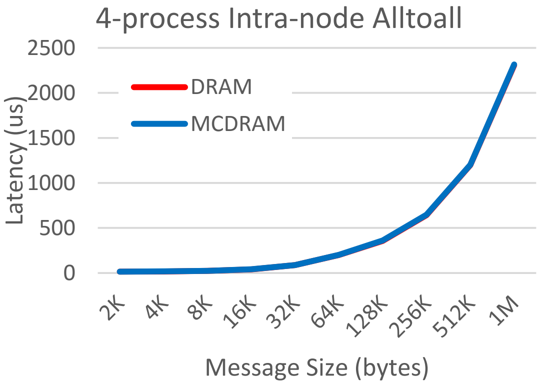
(Based on Colfax International)

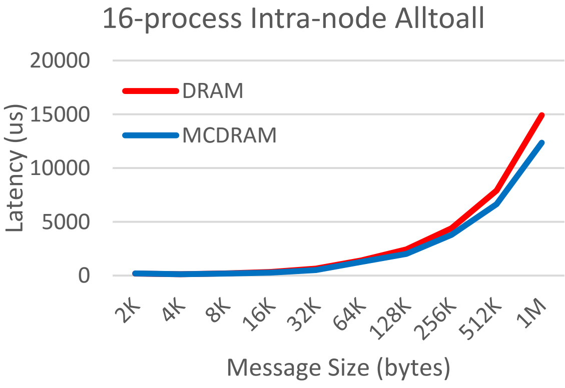
Data Movement on Intel KNL with MCDRAM

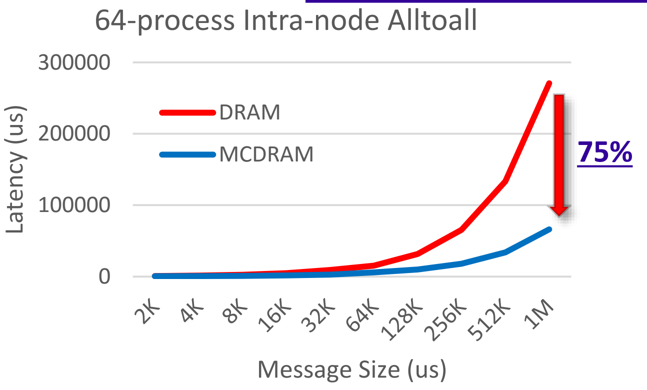
- Regular DRAM and MCDRAM (a High Bandwidth Memory)
- Flexibility for runtimes or users to put data into DRAM or MCDRAM
- Does it lead to performance difference? If so, by how much?
- How does the performance differ with increasing concurrency of data movement?
- What are the power-performance tradeoffs?

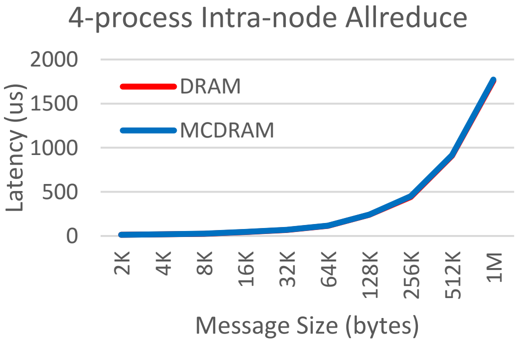
Impact of KNL MCDRAM on MVAPICH2 Collective Performance

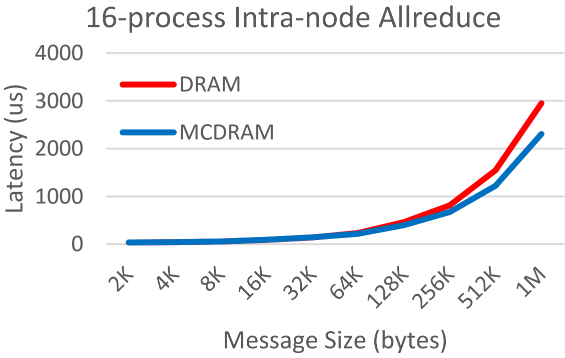
Flat-Quadrant Mode

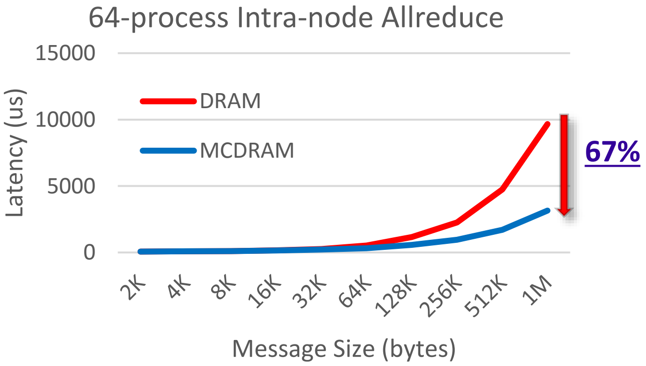

4-process Intra-node Alltoall


16-process Intra-node Alltoall


64-process Intra-node Alltoall


4-process Intra-node Allreduce


16-process Intra-node Allreduce


64-process Intra-node Allreduce

- Benefits of MCDRAM usage seen with larger concurrency/message size
- Similar trends seen in Allgather, Reduce, Bcast, etc.

Looking Ahead …

- MVAPICH2-EA just exploits the knowledge of the current protocols and adopts them for energy savings
- Need new power-aware data movement protocols
  - Build from scratch with energy-consumption as a metric (in addition to performance)
- Need new Parallel Communication Models with Power-Performance trade-offs
  - Similar to LogGP but with power-awareness
  - PLogGP: parameters are functions of voltage (V), frequency (F) and concurrency (C)
    - **Concurrency is a key for HBM**
  - Use PLogGP as building blocks to design data movement protocols and algorithms
- Use these models to design communication library and runtime for MPI+X programming models for next-generation exascale systems

Thank You!

panda@cse.ohio-state.edu

Network-Based Computing Laboratory
http://nowlab.cse.ohio-state.edu/

The MVAPICH2 Project
http://mvapich.cse.ohio-state.edu/

The High-Performance Big Data Project
http://hibd.cse.ohio-state.edu/