David M. Kunzman and Laxmikant V. Kalé, Parallel Programming Laboratory, University of Illinois at Urbana-Champaign

Programming Heterogeneous Systems

Our Goal

To understand how programming models, compilation techniques, and runtime systems can help ease the burden associated with programming heterogeneous systems.

Stepping into the Heterogeneous

Heterogeneous systems are becoming more popular
- Several appearances on the Top500 and Green500 lists
- Can be effective for small research clusters (greater performance per dollar)

Benefits of heterogeneous systems
- High flop rates per dollar
- High flop rates per watt

Tianhe-1A *x86s & GPUs Top500 #1*
Lincoln *x86s & GPUs*
LOEWE-CSC *x86s & GPUs Top500 #22, Green500 #8*
Roadrunner *x86s & Cells Top500 #7*
Nebulae *x86s & GPUs Top500 #3*
TSUBAME 2.0 *x86s & GPUs Top500 #4, Green500 #2+*
QPACE *(x3) x86s & Cells Green500 #5*
Keeneland *x86s & GPUs Green500 #9*
MariCel *POWERs & Cells*
GOSAT-RCF *x86s & GPUs Green500 #10*
Condor *x86s, GPUs, & Cells*

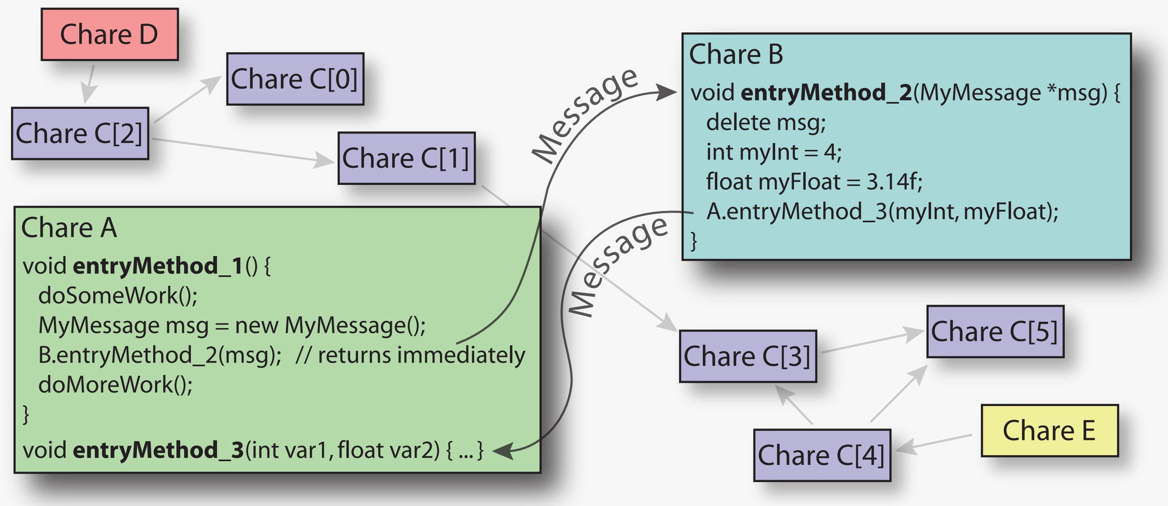
... Here comes the "but" ...

As if parallel programming wasn't already considered hard enough, heterogeneous systems add additional difficulties
- Non-portable, architecture specific code
- Mixture of different execution models (e.g. multicore host with GPU attached)
- Load balancing is harder due to the mismatch in performance characteristics of the various cores

Why Charm++?

Nature of the Charm++ programming model
- Based on asynchronous tasks (unlike threads in MPI)
- Migratable chare objects allow data and computation to be load balanced dynamically by the runtime system
- Highly scalable, mature programming model

Existing Infrastructure
- Runtime system
- Load balancing framework
- Projections (performance visualization tool)



Accelerated Entry Methods

Asynchronous tasks with an arbitrary execution model
- Well defined working sets
- Can execute on either the host or an attached accelerator (based on a load balancing scheme)
- Splits a single standard entry method into two stages
  - Accelerated function body (limited)
  - Associated callback function (general)

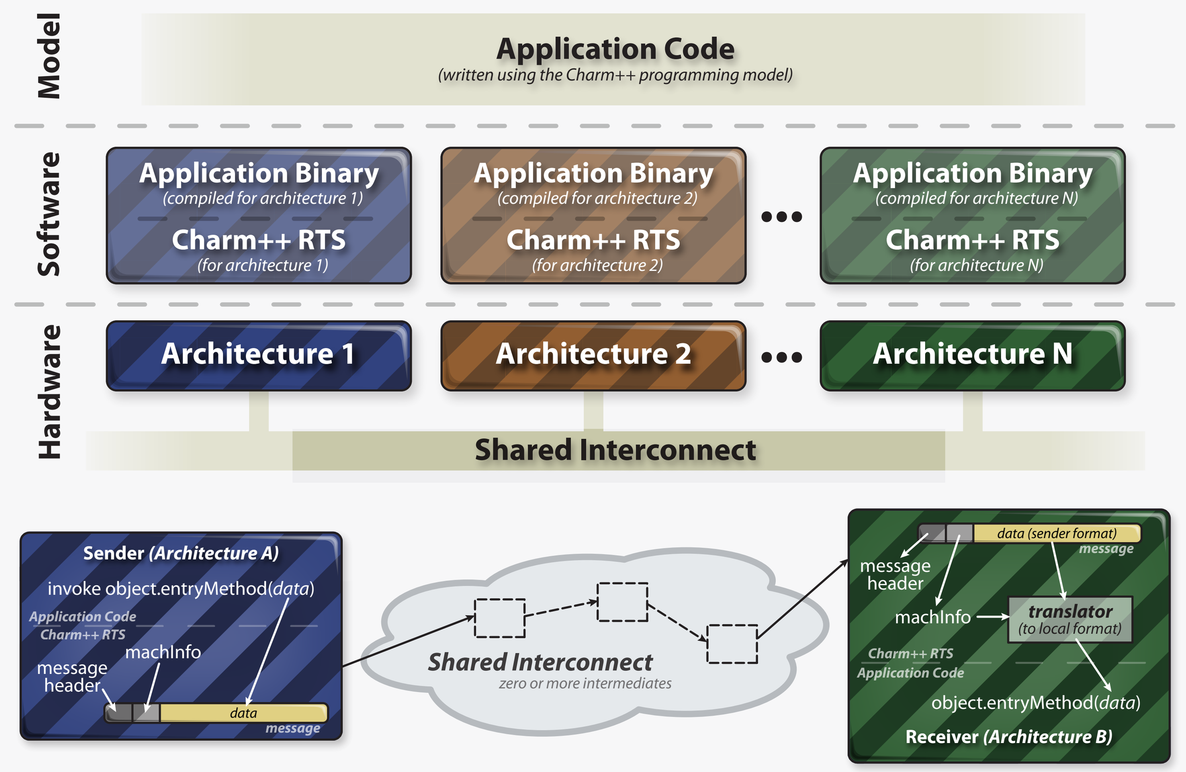
```
entry [ accel ] void dotProduct (
    int N,
    float a[N]
) [
    readOnly : float b[N] <impl_obj->myVector>,
    writeOnly : float c <impl_obj->myResult>
] {
    int i ;
    c = 0.0f;
    for (i = 0; i < N; i++) {
        c += a[i] * b[i];
    }
} dotProduct_callback;

ChareClass::dotProduct_callback() {
    someOtherChare.someEntryMethod(c);
}
```

Passed Parameters (i.e. from caller)
Local Parameters (i.e. data within chare)
Function Body (host or accelerator)
Callback Function (host)

Heterogeneous Execution

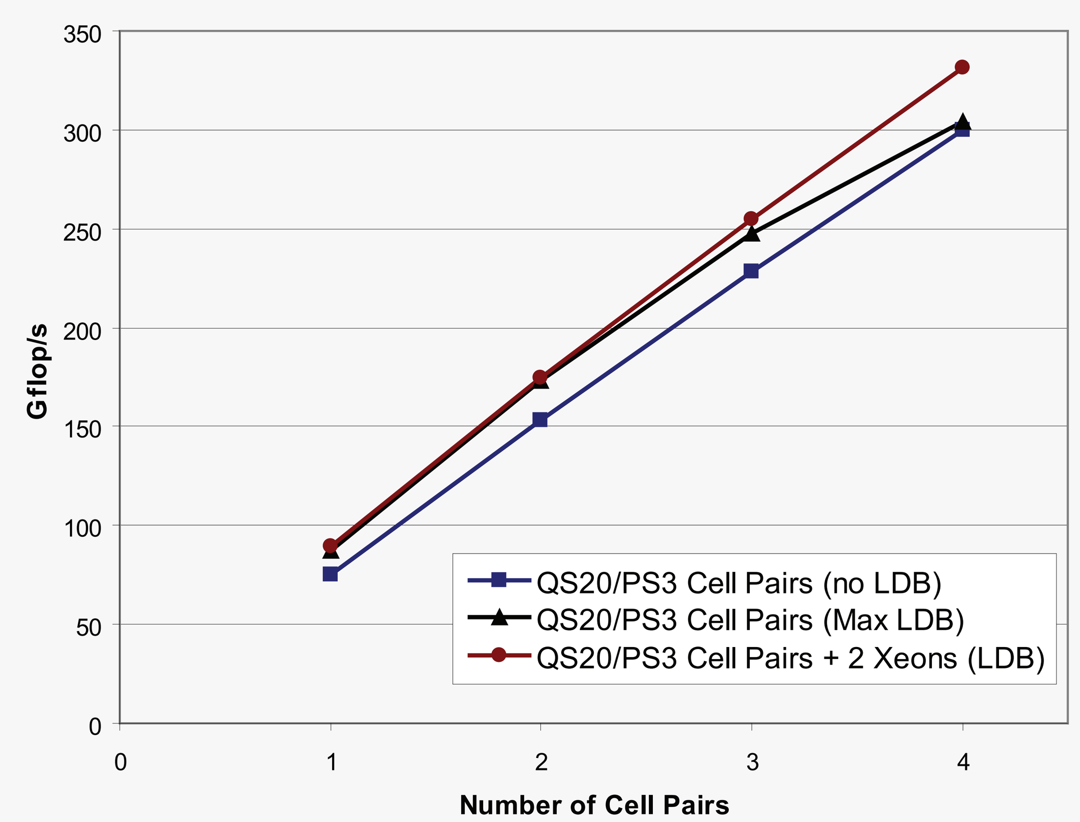
Programming model provides:
- Clearly defined communication boundaries
- Typing and array information (requirements/limitations)

The runtime system handles tedious (but necessary) tasks related to heterogeneous execution, such as:
- Interoperability between different "flavors" of the runtime system within a single application execution
- Real-time manipulation of application data between cores (e.g. big vs. little endian encodings)

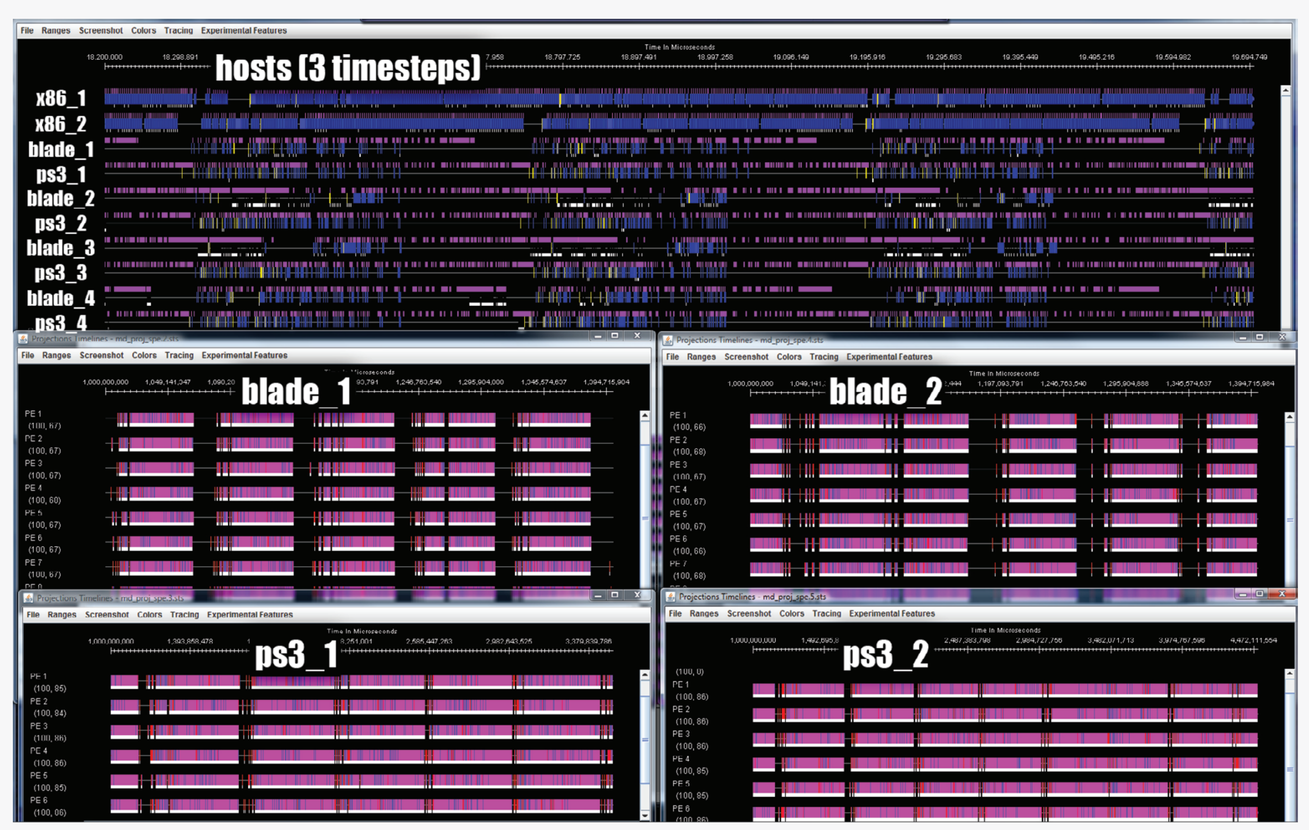


Performance on a Heterogeneous System

Heterogeneous applications may scale better on heterogeneous clusters (compared to a homogeneous cluster)
- We demonstrate a simple MD program that scales better using a mixture of x86 and Cell processors (compared to just using Cells)
  - Reaches 19.8% of peak using one dual-core x86 processor, four PS3 Cells, and four IBM blade Cells
  - Does not include any architecture specific code or code to translate data between architectures
  - Makes use of three different core types (x86, PPE, SPE), three different SIMD extensions, two different memory schemes, and two endian schemes (little and big endian)



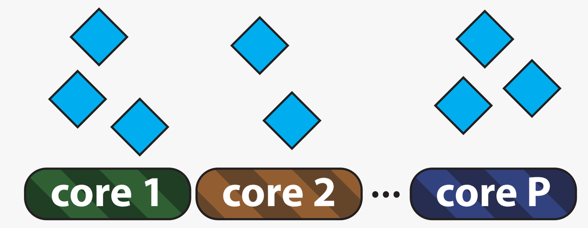
Simple MD program's performance on our cluster (NOTE: 1 Cell Pair = 1 PS3 Cell & 1 QS20 Blade Cell)



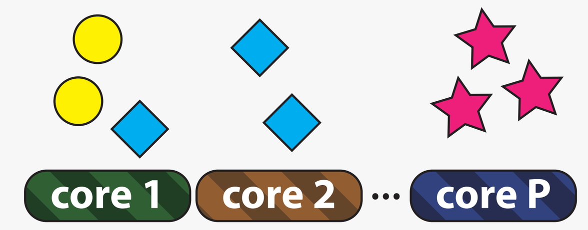
Screenshot from the Projections performance visualization tool being used to show the MD program executing on our heterogeneous cluster.

Continuing & Future Work

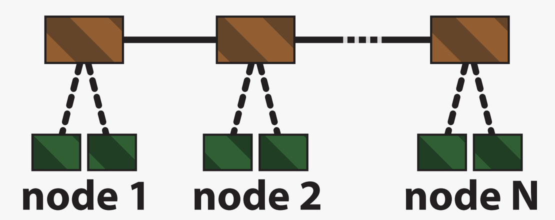
Dynamic Load Balancing



- Dividing a homogeneous workload across a set of heterogeneous cores to minimize the time-to-solution



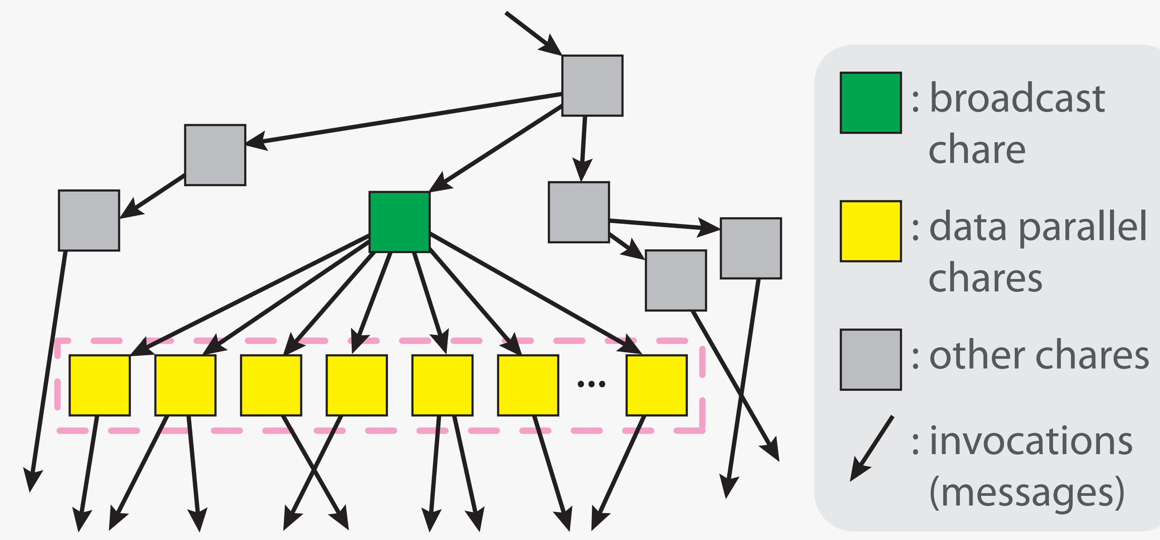
- Dividing a heterogeneous workload across a set of heterogeneous cores, matching sub-computations to the appropriate cores



- Accounting for the non-peer relationships (or topology) of the cores *(i.e. host cores and accelerators are not peers to one another)*

Improved Support for GPUs

The runtime tightens the execution model of sub-computations when it is beneficial
- Start with an arbitrary execution model (asynchronous tasks)
- Take advantage of patterns (broadcasts, stencil patterns, n-body spatial decompositions, etc.)

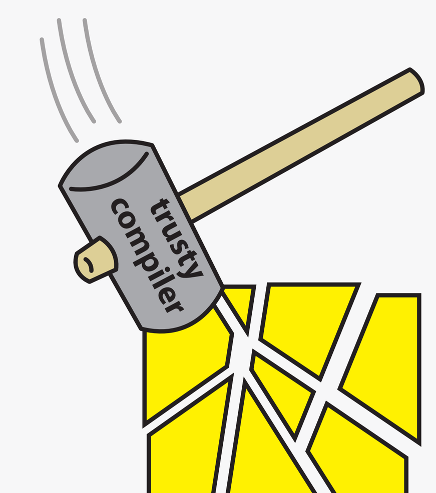


Improved Support for MIC

We are working on it.

Granularity

The compiler assists with automatic granularity adjustments
- Distinguish between communication and computation granularity
- Computation granularity largely based on architecture characteristics
- Compilers focus on what they are good at (small well-defined tasks)
- Programmers focus on what they are good at (identifying high-level parallelism within their applications)



Related Publications:

[1] David M. Kunzman and Laxmikant V. Kale, Programming Heterogeneous Clusters with Accelerators using Object-Based Programming, Journal of Scientific Programming 19 (2011), no. 1, 47–62, IOS Press.
[2] Laxmikant V. Kalé, David M. Kunzman, and Lukasz Wesolowski, Accelerator Support in the Charm++ Programming Model, Scientific Computing with Multicore and Accelerators (Jakub Kurzak, David A. Bader, and Jack Dongarra, eds.), CRC Press (Taylor and Francis Group), December 2010.
[3] David M. Kunzman and Laxmikant V. Kalé. Towards a Framework for Abstracting Accelerators in Parallel Applications: Experience with Cell. In SC '09: Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis, pages 1–12, New York, NY, USA, 2009. ACM.

Poster Created: 5/16/2011

PARALLEL PROGRAMMING LAB PPL UIUC
DEPT. OF COMPUTER SCIENCE, UNIVERSITY OF ILLINOIS

ILLINOIS
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN