Dynamic Adaptive Runtime Systems for Advanced Multipole Methods

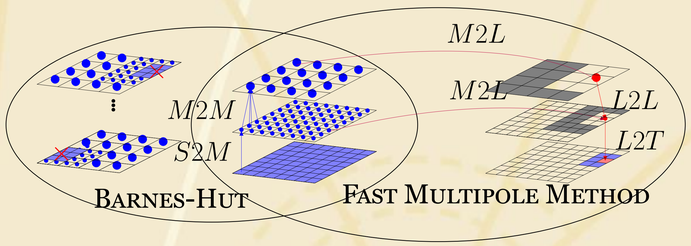
Matthew Anderson, Jackson DeBuhr, Thomas Sterling, Bo Zhang

Introduction

Multipole methods are a key computational kernel for a large class of scientific applications spanning multiple disciplines. Yet many of these applications are strong scaling constrained when using conventional programming practices. This project introduces a new scientific library, DASHMM, built on the ParalleX HPX-5 runtime system, which explores the use of dynamic adaptive runtime techniques to improve scalability and efficiency for multipole-method based scientific computing. DASHMM allows application scientists to rapidly create custom, scalable, and efficient multipole methods, especially targeting the Fast Multipole Method (FMM) and the Barnes-Hut (BH) algorithm.

Scalability Challenges for Multipole Methods[1]

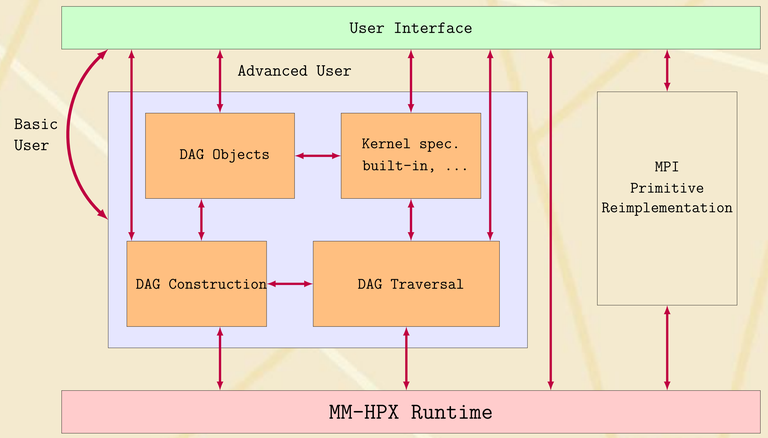
Illustration of the Directed Acyclic Graph (DAG) for the two dimensional FMM and BH algorithm using four levels subdivision. A blue (red) shaded box represents a cluster of source (target) locations while a blue (red) dot represents the multipole (local) expansion.



- Initial placement of the DAG and active load balancing.
- Adjusting the data structure for applications that describe very dynamic physical processes.
- Adoption of mathematical results that reduce the arithmetic complexity and that make overhead more difficult to hide.
- Granularity of parallelism. When applied to Helmholtz kernel for scattering problems, one need to efficiently exploit the parallelism within each basic translation operator of the multipole methods.
- Balance between exposing sufficient parallelism and deviating from the critical path.

Key Components

Overview of DASHMM



Runtime System Approach

We use the following features of the HPX-5 runtime system to address the scalability challenges for the multipole methods.

- Lightweight, ephemeral threads, allowing a finer-grained parallelism than traditional programming models.
- Parcels, a form of active message, allow the runtime to express message-*driven* computation, allowing work to move to data in addition to the more traditional approach where data moves to work.
- Local Control Objects (LCO) are event driven conditional structures that cause a thread or some other action to be instantiated once a condition is met. LCOs encode the dependency structure of the data-flow computation.
- Enqueue parcels on the LCOs for a completely data-driven implementation that reveals the full DAG to the runtime, allowing the runtime to
  - Monitor future system loads
  - Relocate data/task actively
  - Cache data on different localities

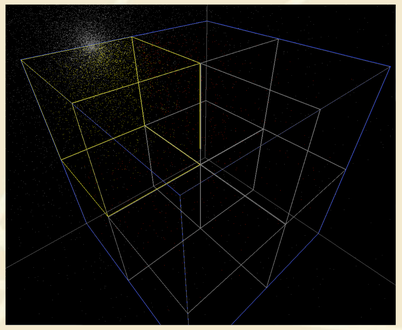
Current Progress

- Completed draft of basic user interface; implementation has begun.

DASHMM basic user interface

```
dashmm_init();
dashmm_evaluate(method, kernel, accuracy, source, target, stats);
dashmm_finalize();
```

- Completed implementation of the Laplace kernel that incorporates the additional exponential expansion mechanism to reduce the arithmetic complexity. Implementing the Yukawa kernel and low-frequency Helmholtz kernels.
- Investigating choices of kernel-independent FMMs for end-of-project delivery.
- Implemented stand-alone Barnes-Hut code in HPX-5, which serves the starting point for DASHMM methods and architecture improvements.
- Performed case studies for various components of the DASHMM system.
- Created visualization tool Lorax.



New features planned include visualizing execution and scheduling history (see website for use video).

- `https://www.crest.iu.edu/projects/dashmm/`

[1] *Asynchronous Task Scheduling of the Fast Multipole Method using various Runtime Systems, DFM 2014*

INDIANA UNIVERSITY BLOOMINGTON
SCHOOL OF INFORMATICS AND COMPUTING