ExaCT CENTER FOR EXASCALE SIMULATION OF COMBUSTION IN TURBULENCE

ExaCT Programming Models/Environment Activities

http://exactcodesign.org

Pat McCormick (LANL)

Programming Models Lead

2015 ECI PM/E Workshop
March 9 – 11, 2015
Rockville, MD

U.S. DEPARTMENT OF ENERGY Office of Science · Sandia National Laboratories · Los Alamos National Laboratory · NREL · Berkeley Lab · Rutgers · The University of Texas at Austin · Oak Ridge National Laboratory · Lawrence Livermore National Laboratory · Georgia Tech · Stanford University · The University of Utah

Overview

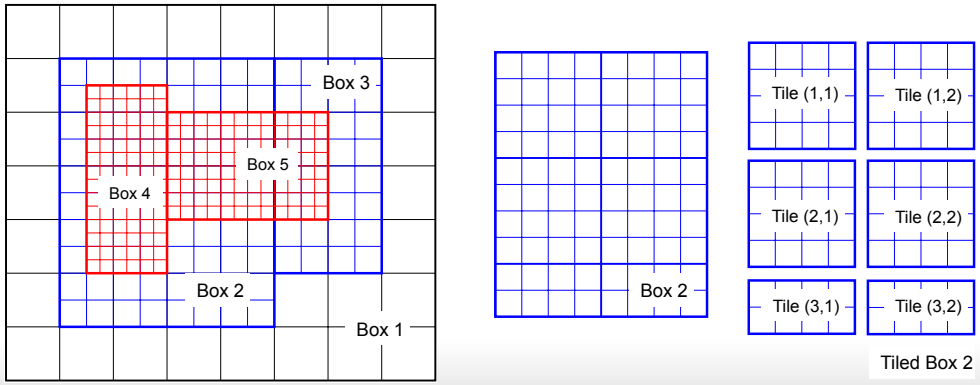
- **Activities cover: Node-level, full-system and modeling**
  - *Node Level*: TiDA – Tiling Abstractions To Manage Data Locality
  - *Modeling*: Rambutan – Asynchronous Runtime Modeling Tool
  - *Full-system*: Legion – Programming model for heterogeneous, distributed machines
- *Exploring only proxy-applications is not sufficient…*

Stanford	LBL	LANL	LLNL
Michael Bauer	John Shalf	Pat McCormick	Dan Quinlan
Wonchan Lee	Cy Chan	Sam Gutierrez	Pei-Hung Lin
Sean Treichler	Didem Unat	Galen Shipman	Chunhua Liao
Elliott Slaughter	Sam Williams	Christine Sweeney	
Zhihao Jia		Dean Prichard	
Alex Aiken			

TiDA: Tiling Abstraction for Improved Node Performance

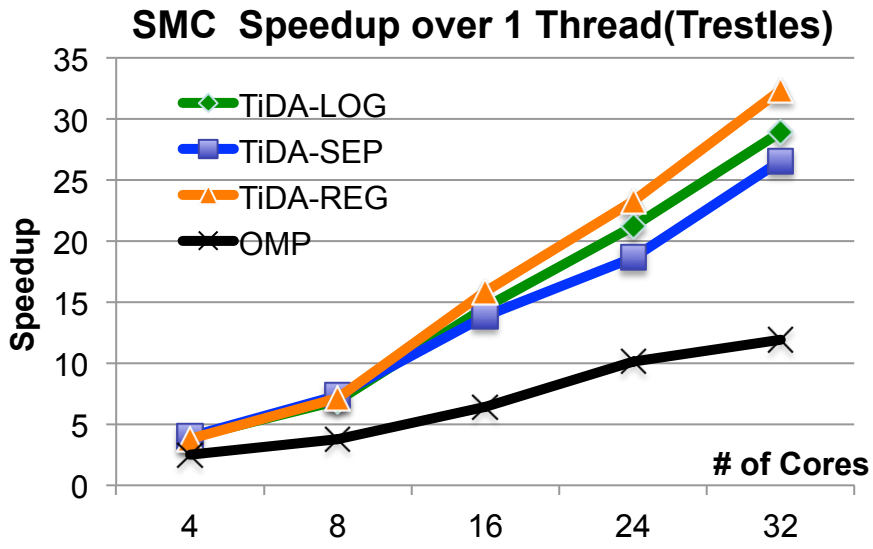
Didem Unat, John Shalf, John Bell

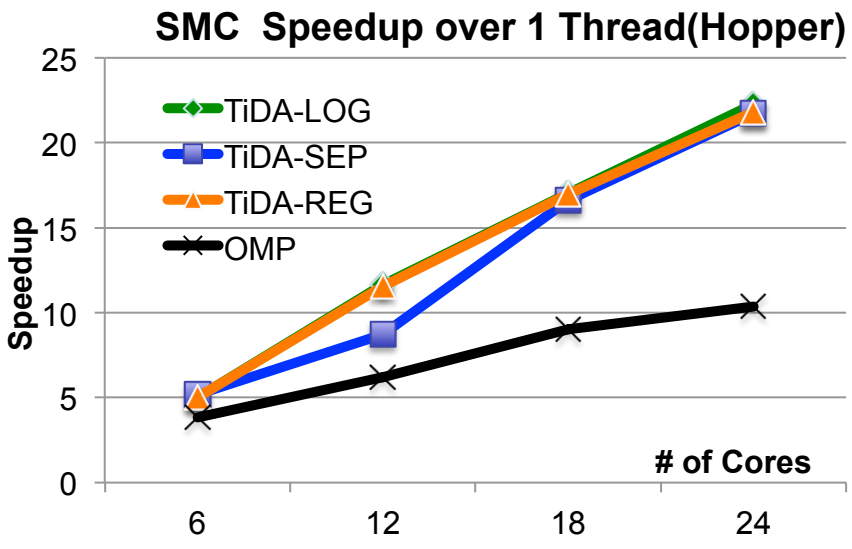
- **Most tiling optimizations are static loop transformations – *inadequate for adaptive codes***
  - No/little communication between compiler and runtime
  - *Tiling should be decoupled from loops and elevated to the programming model*
- **Tiles become the fundamental unit of work and the core concept for data locality**
  - Coarse-grained parallelism across tiles, fine-grained (vectorization & instruction ordering) within a tile



What Problem is TiDA/Tiling Abstraction Solving?

- **NUMA malloc routines also tedious *(automate)***
- **Tedious and verbose OMP directives – error prone *(automate)***
- **Thread pinning is tedious (automate it)**
- **Abstracts data layout choices between GPU and CPU (independent tiles with ghost regions vs. shared tiles)**





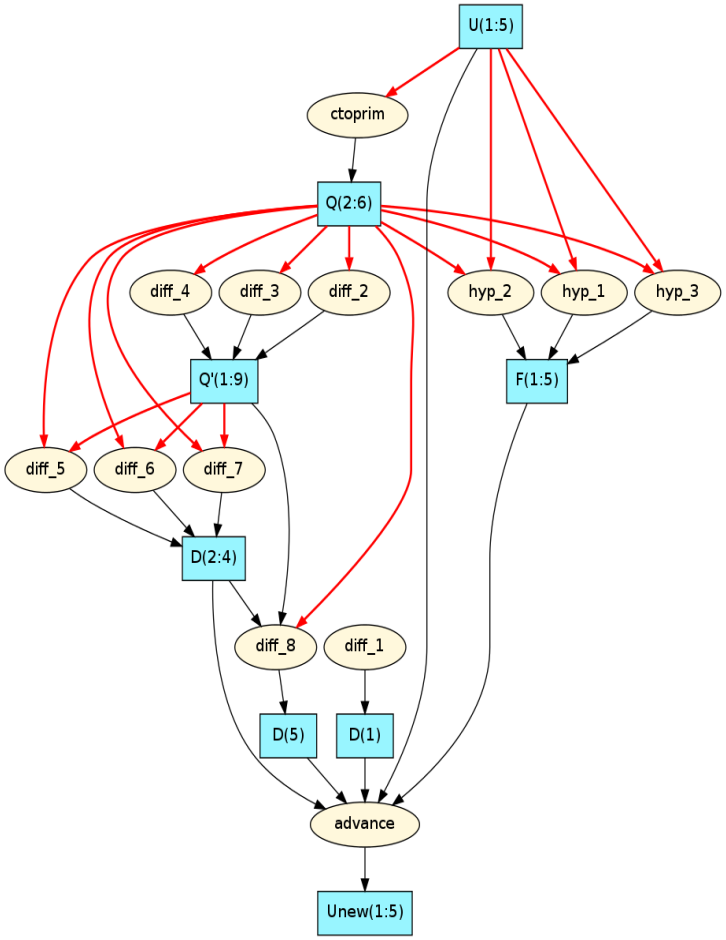
TiDA: Summary

- **Similar examples in library and DSL form**
  - HTA: Hierarchical Tiled Arrays
  - RAJA & KOKKOS: C++ template meta programming *(many others!)*
- **All arrived at similar underlying concepts**
  - Lamba functions to relax loop nest order
  - Abstracts data physical layout from logical layout
  - Perhaps we have found a reasonable solution?
- **For Tiling Abstractions, see PADAL (Programming Abstractions for Data Locality)**

http://www.padalworkshop.org/

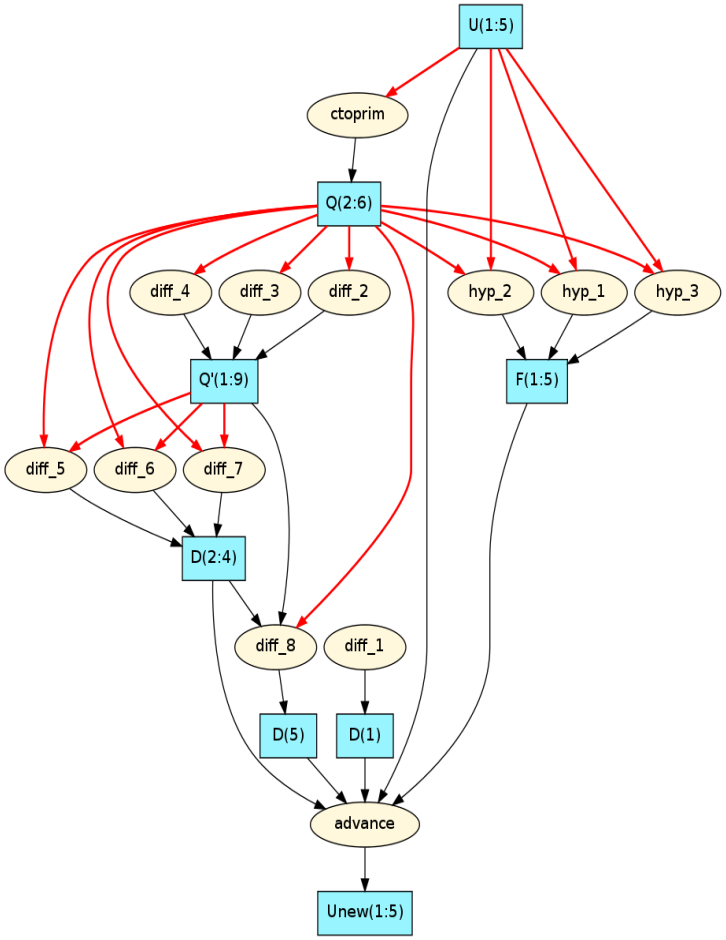
Rambutan: Asynchronous Runtime Modeling Tool

Cy Chan, John Bachan, John Shalf



- **Why do we need a runtime modeling tool?**
  - Better understand when a task-driven, asynchronous system beneficial
- **Many factors :**
  - Algorithm characteristics
  - Machine characteristics
  - Programming and execution model
- Production runtimes are complex and difficult to analyze
- Consider future architecture designs

Rambutan: Asynchronous Runtime Modeling Tool



- **Explore the impact of various design choices:**
  - Categorize/measure task overheads
  - Model irregular algorithms and hardware features (throttling, near-threshold voltages, etc.)
  - Estimate impact of hardware support for task management
  - Insight into task decomposition, granularity, and scheduling strategies

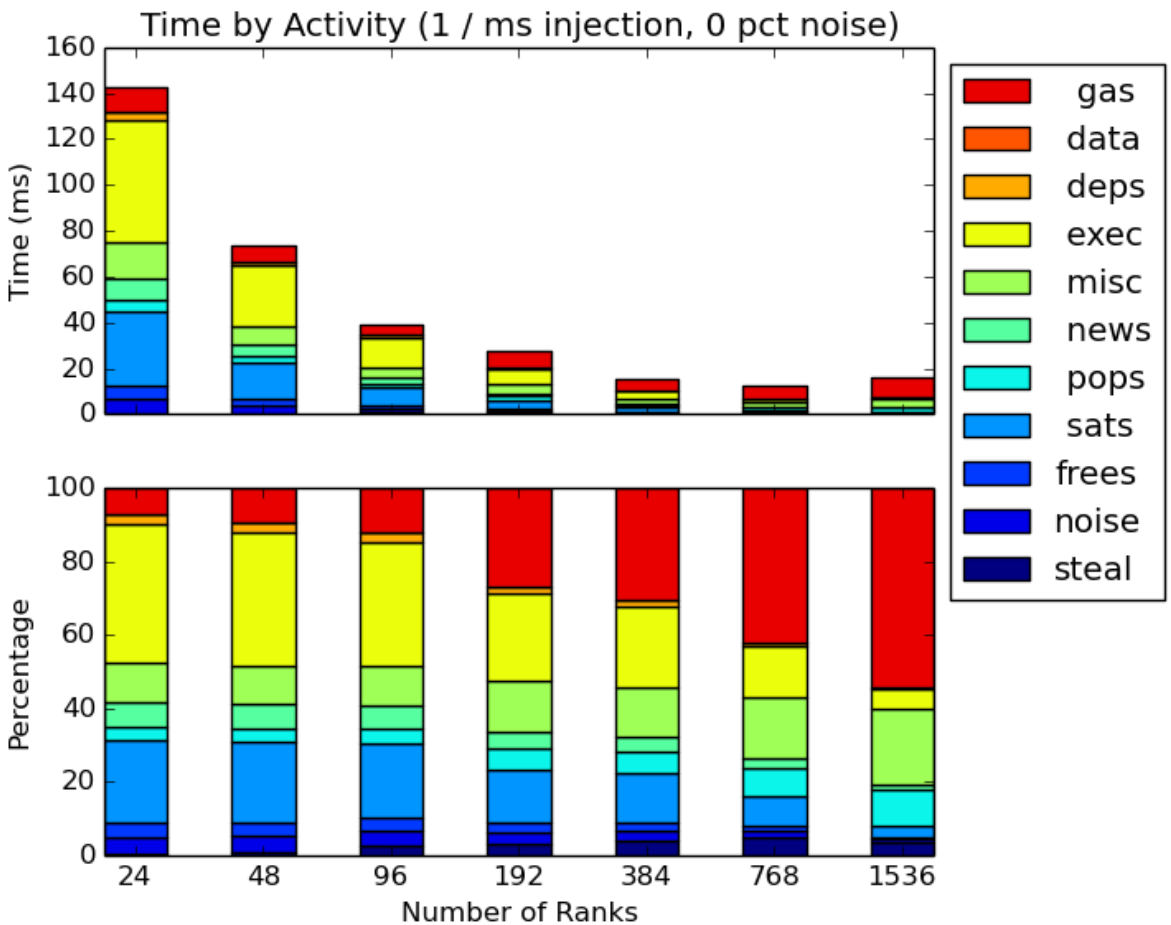
Rambutan Instrumentation

- **Timers added to capture time spent within both the app's registered callbacks and the runtime**

Type	Operation	Description
Application-Specific	Registration	Task and data dependencies
	Execution	Computational work
	Finalization	Compute and satisfy dependents
Task Management	New / Delete	Allocate and free tasks
	Queue / Dequeue	Move tasks between queues
	Satisfaction	Satisfy task dependencies
	Load Balancing	Balance tasks across ranks
Data Management	Fetch	Get remote data for reads
	Commit	Write back dirty remote data

Rambutan: UTS Benchmark

- **Unbalanced Tree Search (UTS) with 4 million nodes**



Legion Programming Model

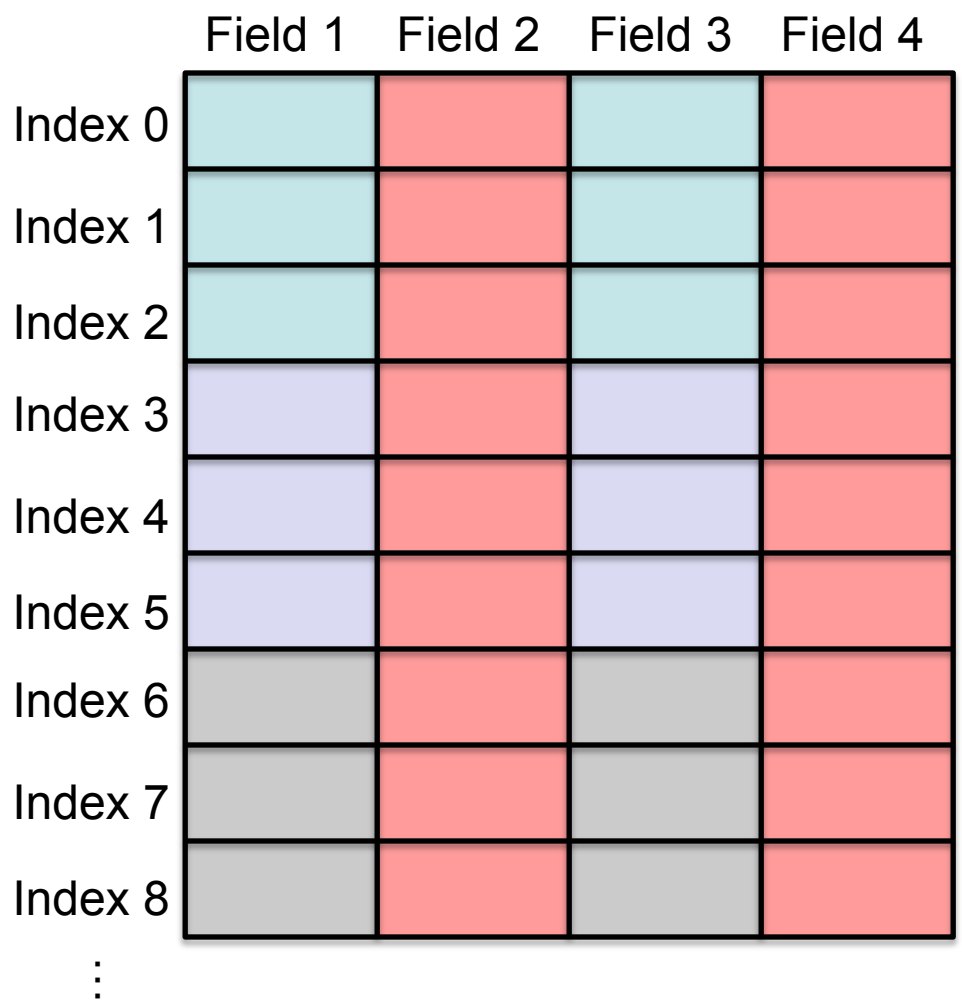
Michael Bauer , Wonchan Lee, Sean Treichler, Elliott Slaughter, Zhihao Jia, Alex Aiken, Pat McCormick, Sam Gutierrez, Galen Shipman, Hemanth Kolla, Ankit Bhagatwala, Ramanan Sankaran, Jackie Chen

- **Task-based model targeted at heterogeneous, distributed-memory machines**
  - Large spread, and variability, of communication latencies
  - Goal: Latency tolerant
- **Data-centric – focus on giving the runtime knowledge about the structure of data ("Logical Regions")**
- **Separation of concerns – "mapping" of tasks and data movement (locality + affinity) independent of task implementation**

Data Model – Logical Regions

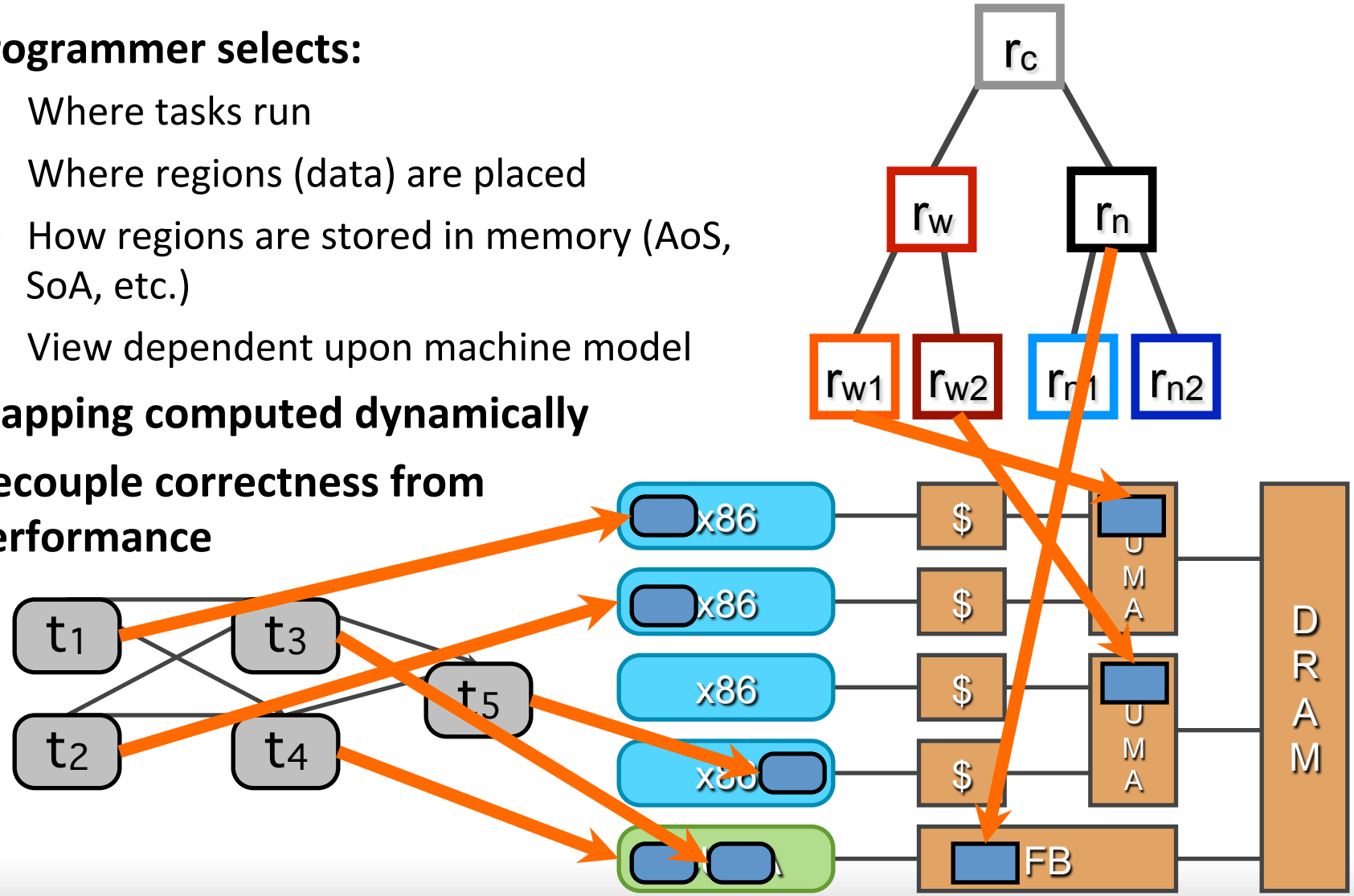
- Unbounded set of rows (index space – unstructured, 1D, 2D, …)
- Bounded set of columns (fields)
- Can be partitioned (along index space)

- **Tasks operate on regions**
  - Must specify which fields and how they "use" them (fields and privileges: read, write, read +write, exclusive, etc.)
  - Allows fields to be "sliced"
- Tasked "launched" in program order (execution order relaxed based on dependencies)



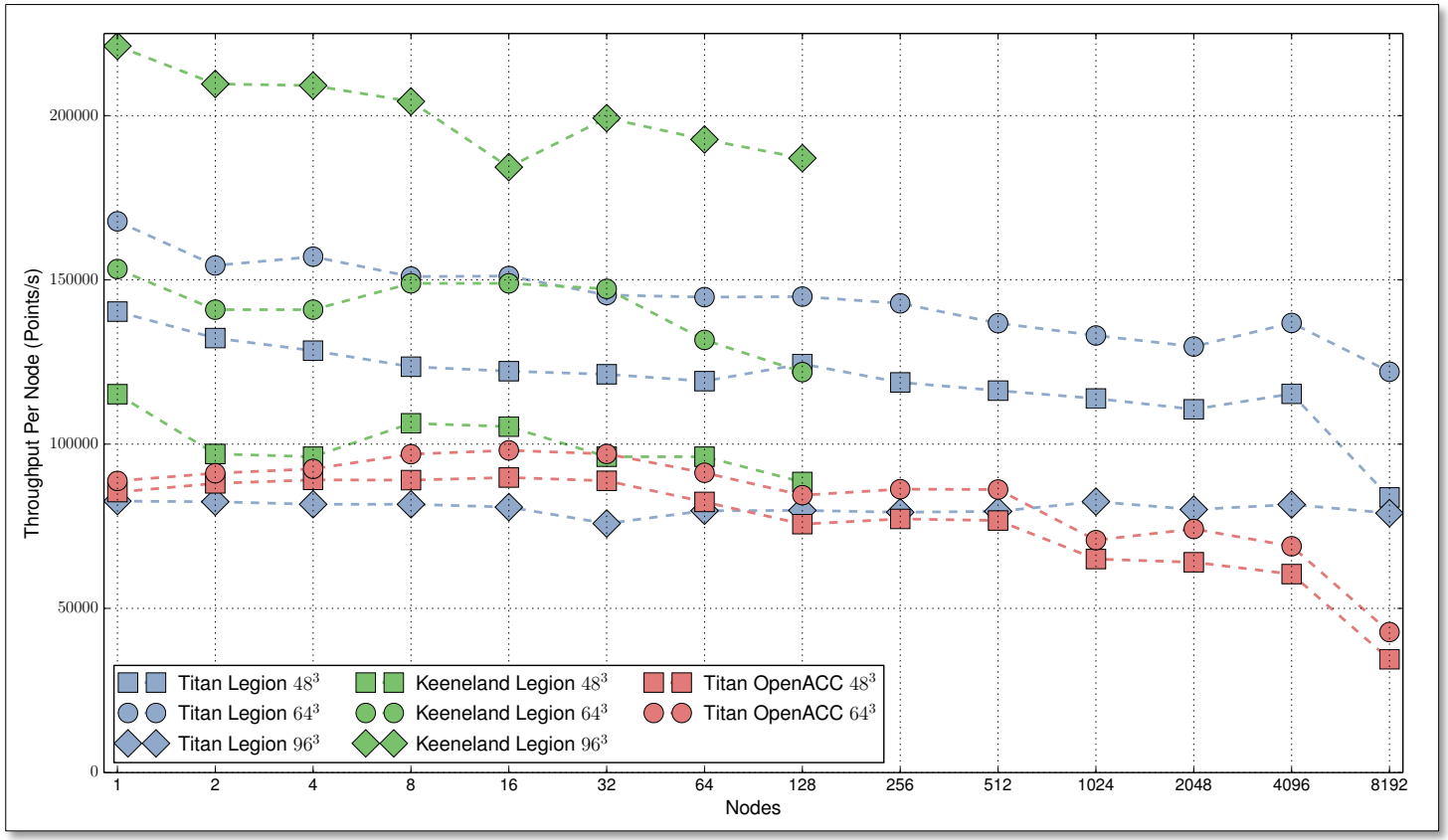
Mapping Interface

- **Programmer selects:**
  - Where tasks run
  - Where regions (data) are placed
  - How regions are stored in memory (AoS, SoA, etc.)
  - View dependent upon machine model
- **Mapping computed dynamically**
- **Decouple correctness from performance**



Legion S3D

- Ported production combustion simulation
  - ~200K lines of Fortran
  - Direct numerical simulation using explicit methods



Heptane:
1.73X - 2.85X faster between 1024 and 8192 nodes

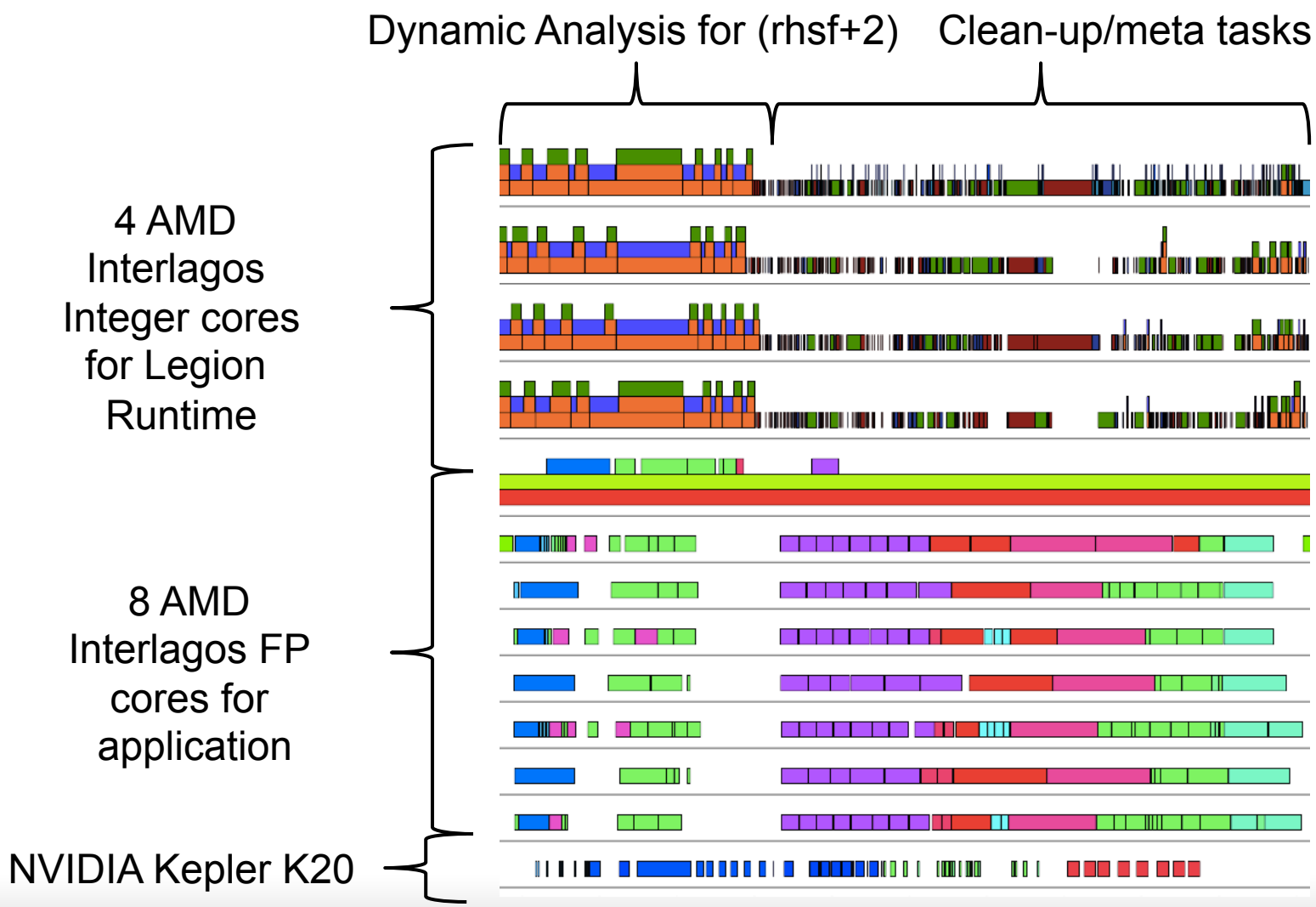
Runtime-based implementation

MPI+Legion structure

New science!

But… This is the runtime programmed directly!

Mapping for Heptane 48^3



More Legion Information

- Web site: http://legion.stanford.edu
  - Open source repository
  - Legion Bootcamp, Dec. 2014
    - ~70 attendees from DoE, industry, Stanford
    - All materials available on-line (see "bootcamp" tab)

Summary

- **Goal: Put all the pieces together…**
  - Legion – internode tasking model
  - TiDA – intranode cache-/NUMA-aware model (e.g. used within Legion's tasks)
  - Rambutan – Analysis/modeling of task-based performance tradeoffs (connection to Legion mappers & potential architecture design decisions)

Thank You

Questions?