Legion: Programming Distributed Heterogeneous Architectures with Logical Regions
Michael Bauer, Sean Treichler, Elliott Slaughter, Alex Aiken
Stanford University

Overview

Three Important Trends
1. Cost of Data Movement Dominates
2. Dynamism in Both Hardware and Software
3. Heterogeneity of Processors and Memory

Legion Goals
1. Abstractions for logically describing program data to minimize data movement
2. Emphasize runtime decision making to handle dynamism
3. Decouple specification from mapping to handle heterogeneity

Logical Regions: A Relational Abstraction for Data

Logical regions describe data in a relation style: entries (rows) and fields (columns).
- Support some relational operators for providing different views on data.
- Partitioning is a relation (p)
- Field-slicing is projection (σ)
- Locality: elements in same logical regions
- Independence: disjoint logical regions in same partition
- Privilege: non-interference: performing non-interfering access (e.g. both read-only)

Logical regions can encode arbitrary data structures.

Implicit Task Parallelism

Tasks are the unit of computation in Legion
- Tasks name logical regions and fields they will access at dispatch
- Each logical region and field annotated with privilege: read-only, read-write, reduce
- Tasks can recursively launch arbitrary sub-tasks for nested parallelism

Tasks issued in program order, parallelism inferred from non-interference
- Sub-region non-interference: accessing disjoint sub-regions
- Field non-interference: accessing disjoint sets of fields
- Privilege non-interference: performing non-interfering access (e.g. both read-only)

S3D

Production combustion simulation from the Department of Energy
- Full scale application consisting of more than 200K lines of Fortran MPI
- Significant task and data level parallelism
- Data movement is the limiting factor

Ported main simulation loop (100K lines) of S3D into Legion C++
- Interoperate with MPI version
- Legion automatically discovers significantly task level parallelism
- Can explore different mapping running on Titan, world’s #2 supercomputer

Mapping

Legion applications are mapped onto different architectures
- Tasks assigned to processors, region instances assigned to memories
- Sub-region: parent and child logical regions
- Field-slicing: disjoint sets of fields
- Privilege: non-interference: performing non-interfering access (e.g. both read-only)

Sub-Region Non-Interference Field Non-Interference Privilege Non-Interference

Legion runtime operates similarly to a hardware out-of-order processor

Evaluation

Many applications ported to Legion, showing both strong and weak scaling
- Circuit simulation, AMR, fluid flow, unstructured mesh, S3D
- Can handle CPU+GPU, Infiniband, Gemini, Aires, NUMA, ...
- Legion automatically discovers significantly task level parallelism
- Full scale application consisting of more than 200K lines of Fortran MPI

S3D performance results demonstrate that Legion can weak scale
- Compare against MPI+OpenACC version tuned by NVIDIA and Cray teams
- Take best Legion mapping after trying many different tuning techniques
- Between 2.0 and 2.27X faster than MPI+OpenACC

Throughput (Points/s)

S3D

Legion vs MPI+OpenACC

Non-Interference

Dep. Analysis Map Distribute Execute Resolve Spec. Complete Commit

Legion runtime operates similarly to a hardware out-of-order processor

Partitioning

Logical regions can be partitioned into sub-regions
- Partitions are computed dynamically, either disjoint or aliased
- Logical regions can be partitioned recursively
- Logical regions can be partitioned into sub-regions

Logical region trees capture important properties of program data
- Independence: disjoint logical regions in same partition
- Locality: elements in same logical regions
- Sub-region: parent and child logical regions
- Aliasing: aliased partitions and different partitions of same region

Mapping decisions are independent of correctness
- Makes tuning Legion applications simple
- Parting can be performed easily and only requires writing a new mapper

Mappers are customisable and composable
- Provide a default mapper with heuristics
- Can write Legion applications and gradually refine mapping by overriding virtual functions
- Different mappers in same application

Heptane Mechanism

DME Mechanism