Chapel: Project Overview
Outline

- What we do
- Project Status
- Who we are
Chapel Work

• Chapel Team’s Focus:
  • **specify Chapel** syntax and semantics
  • **implement open-source prototype compiler** for Chapel
  • **perform code studies** of benchmarks, apps, and libraries in Chapel
  • **do community outreach** to inform and learn from users/researchers
  • **support collaborators and users** of code releases
  • **refine** language based on all these activities
In a nutshell:

- Most features work at a functional level
- Many performance optimizations remain

This is a good time to:

- Try out the language
- Give us feedback to improve the language
- Use Chapel for parallel programming education
- Use Chapel for non-performance-critical projects

In evaluating the language:

- Try to judge it by how it should *ultimately* perform rather than how it does today
  - lots of low-hanging fruit remains, as well as some challenges
“I Like Chapel, how can I help?”

- Let people know that you like it and why
  - your colleagues
  - your employer/institution
  - Cray leadership

- Help us evolve it from prototype to production
  - contribute back to the source base
  - collaborate with us
  - help fund us to grow the team
  - help us get from “How will Cray make Chapel succeed?” to “How can we as a community make Chapel succeed?”
Join Our Team

• Cray:
  Brad Chamberlain  Sung-Eun Choi  Greg Titus  Lee Prokowich  Vass Litvinov

• External Collaborators:
  Albert Sidelnik  Jonathan Turner  Srinivas Sridharan

• Interns:
  Jonathan Claridge  Hannah Hemmaplardh  Andy Stone  Jim Dinan  Rob Bocchino  Mack Joyner

You?
Currently:
• Jr. SW Eng. doing R&D on targeting next-generation nodes
  • GPUs, tiled architectures, scratchpad memories, manycore, ...

Upcoming:
• Hopefully more this year
Select Collaborations

- **ORNL/Notre Dame** (Srinivas Sridharan, Jeff Vetter, Peter Kogge): Asynchronous software transactional memory over distributed memory
- **UIUC** (David Padua, Albert Sidelnik, Maria Garzarán): CPU-GPU computing
- **Sandia** (Kyle Wheeler, Rich Murphy): Chapel over Qthreads user threading
- **BSC/UPC** (Alex Duran): Chapel over Nanos++ user-level tasking
- **Argonne** (Rusty Lusk, Rajeev Thakur, Pavan Balaji): Chapel over MPICH
- **CU Boulder** (Jeremy Siek, Jonathan Turner): Interfaces, concepts, generics
- **U. Oregon/Paratools Inc.** (Sameer Shende): Performance analysis with Tau
- **U. Malaga** (Rafael Asenio, Maria Gonzales, Rafael Larossa): Parallel file I/O
- **PNNL/CASS-MT** (John Feo, Daniel Chavarria): Cray XMT tuning
- **(your name here?)**
Collaboration Ideas (see chapel.cray.com for more details)

- memory management policies/mechanisms
- dynamic load balancing: task throttling and stealing
- parallel I/O and checkpointing
- exceptions; resiliency
- language interoperability
- application studies and performance optimizations
- index/subdomain semantics and optimizations
- targeting different back-ends (LLVM, MS CLR, ...)
- runtime compilation
- library support
- tools: debuggers, performance analysis, IDEs, interpreters, visualizers
- database-style programming
- (your ideas here...)

Chapel: Project Overview
Chapel Team’s Next Steps

- Expand our set of supported distributions
- Continue to improve performance
- Continue to add missing features
- Expand the set of codes we are studying
- Expand the set of architectures we are targeting
- Support the public release
- Continue to support collaborations and seek out new ones
- Continue to expand our team
Questions?