What is Chapel?

Chapel: A modern parallel programming language

• portable & scalable
• open-source & collaborative

Goals:

• Support general parallel programming
• Make parallel programming at scale far more productive
Why Consider New Languages at all?

- **Syntax**
  - High level, elegant syntax
  - Improve programmer productivity

- **Semantics**
  - Static analysis can help with correctness
  - We need a compiler (front-end)

- **Performance**
  - If optimizations are needed to get performance
  - We need a compiler (back-end)

- **Algorithms**
  - Language defines what is easy and hard
  - Influences algorithmic thinking

[Source: Kathy Yelick, CHIUW 2018 keynote: Why Languages Matter More Than Ever]
A Year in the Life of Chapel

• Two major releases per year (Spring & Fall)
  • ~a month later: detailed release notes
  • latest releases: Chapel 1.21 / 1.22, released April 9<sup>th</sup> / 16<sup>th</sup> 2020

• CHIUW: Chapel Implementers and Users Workshop (May/June)
  • typically co-located with IPDPS or PLDI

• SC (November)
  • annual CHUG (Chapel Users Group) happy hour
  • some years: talks, tutorials, panels, BoFs, posters, …

• Development, user support, talks, research visits, social media, … (year-round)
A Year in the Life of Chapel

• Two major releases per year (Spring & Fall)
  • ~a month later: detailed release notes
  • latest releases: Chapel 1.21 / 1.22, released April 9\textsuperscript{th} / 16\textsuperscript{th} 2020

• CHIUW: Chapel Implementers and Users Workshop (May/June)
  • typically co-located with IPDPS or PLDI

• SC (November)
  • annual CHUG (Chapel Users Group) happy hour
  • some years: talks, tutorials, panels, BoFs, posters, …

• Development, user support, talks, research visits, social media, … (year-round)
CHIUW 2014–2019: Professional Edition*

* = actually this photo wasn’t taken at CHIUW...
CHIUW 2020: Live from our Bedrooms Home Offices

• Due to Covid-19, this is the first time CHIUW has been held online
• We’re grateful to all the speakers who agreed to participate remotely, spanning:
  • 3 continents
  • 4 countries
  • 5 time zones (?)
  • 11 organizations
• Roughly half of today’s talks will be pre-recorded, the other half will be live
  • All will feature live, post-talk Q&A sessions
  • Talk slides are available on the CHIUW website:
CHIUW 2020: Technical Details

• Ask questions to the speakers using Zoom’s Q&A tool
  • Session chairs will help relay the questions to speakers
• Use the chat window for technical help / questions
  • Or to chat socially during breaks and pauses
• Please use professionalism and common sense in all cases
CHIUW 2020: Organizing Committee

General Chair:
- Benjamin Robbins, Cray/HPE

Steering Committee:
- Michael Ferguson, Cray/HPE
- Mike Merrill, U.S. DOD
- Nikhil Padmanabhan, Yale University
- Marcin Zalewski, NVIDIA

Program Committee:
- Brad Chamberlain (chair), Cray/HPE
- Cathie Olschanowsky (co-chair), Boise State University
- Maryam Dehnavi, University of Toronto
- Clemens Grelck, University of Amsterdam
- Paul H. Hargrove, Lawrence Berkeley National Laboratory
- Engin Kayraklioglu, Cray/HPE
- Milind Kulkarni, Purdue University
- Josh Milthorpe, Australian National University
- Tyler Simon, UMBC
- Christian Terboven, RWTH Aachen University
- Rich Vuduc, Georgia Tech
- Marcin Zalewski, NVIDIA

pre: Chapel 101 (optional)

now: Welcome, State of the Project

9:15: talks: Chapel Language Evolution

10:20: talks: Arkouda, including our keynote talk (next slide)

12:00: talks: Applications of Chapel

1:55: talks: Chapel on GPUs

2:50: talks: Implementing Chapel

3:35: Open Discussion Session

future: Online Coding Day?
Arkouda: Chapel-Powered, Interactive Supercomputing for Data Science

Dr. William Reus, U.S. DOD

Abstract (excerpts): …This talk motivates and demonstrates the interactive use of up to hundreds of HPC nodes in data science workflows with an open-source package called Arkouda, which exposes massively parallel, distributed NumPy-like arrays to a Jupyter notebook running Python 3.

…the computational heart of Arkouda is a relatively compact yet highly scalable Chapel interpreter that implements a powerful set of data science primitives.

…the competitive advantage of Arkouda comes from the unique position Chapel holds as a productive language with performance and scaling on par with industry-standard HPC technologies.
CHIUW 2020: Technical Talks

• Chapel Language Evolution
• Arkouda
• Applications of Chapel
• Chapel on GPUs
• Implementing Chapel
CHIUW 2020: Technical Talks

• Chapel Language Evolution
  Towards Stability in the Chapel Language
  Michael Ferguson, Cray, a Hewlett Packard Enterprise Company
  Visibility Control: Use and Import Statement Improvements
  Lydia Duncan, Cray, a Hewlett Packard Enterprise Company

• Arkouda

• Applications of Chapel

• Chapel on GPUs

• Implementing Chapel
CHIUW 2020: Technical Talks

• Chapel Language Evolution

• Arkouda

  Arkouda: Chapel-Powered, Interactive Supercomputing for Data Science
  Bill Reus, U.S. DOD

  Squeezing Performance out of Arkouda
  Elliot Ronaghan, Cray, a Hewlett Packard Enterprise Company

• Applications of Chapel

• Chapel on GPUs

• Implementing Chapel
CHIUW 2020: Technical Talks

• Chapel Language Evolution

• Arkouda

• Applications of Chapel
  
  Development of Parallel CFD Apps on Distributed Memory with Chapel
  Matthieu Parenteau, Polytechnique Montreal
  
  Computing Hypergraph Homology in Chapel
  Jesun S. Firoz, Pacific Northwest National Laboratory
  
  Exploring Chapel Productivity Using Some Graph Algorithms
  Richard F. Barrett, Sandia National Laboratories
  
  Simulating Ultralight Dark Matter in Chapel
  Nikhil Padmanabhan, Yale University

• Chapel on GPUs
CHIUW 2020: Technical Talks

- Chapel Language Evolution
- Arkouda
- Applications of Chapel
- Chapel on GPUs
  - Exploring a Multi-Resolution GPU Programming Model for Chapel
    Akihiro Hayashi, Georgia Institute of Technology
  - Chapel on Accelerators
    Rahul Ghangas, The Australian National University
- Implementing Chapel
CHIUW 2020: Technical Talks

• Chapel Language Evolution
• Arkouda
• Applications of Chapel
• Chapel on GPUs
• Implementing Chapel

Paving the way for Distributed Non-Blocking Algorithms and Data Structures in the Partitioned Global Address Space model
Garvit Dewan, Indian Institute of Technology, Roorkee

An Automated Machine Learning Approach for Data Locality Optimizations in Chapel
Engin Kayraklioglu, Cray, a Hewlett Packard Enterprise company
State of the Chapel Project
Major Efforts since CHIUW 2019

• Three New Major Releases:
  • Chapel 1.20.0: September 19, 2019
  • Chapel 1.21.0: April 9, 2020
  • Chapel 1.22.0: April 16, 2020 (essentially 1.21.0 with 0-based indexing)

• Primary Focus Areas:
  • Language Stability ("Chapel 2.0")
  • Performance and Scalability
  • Interoperability, particularly with Python
  • Portability, particularly to Cray Shasta for early access customers
Chapel 2.0
Chapel 2.0: Concept

• It seems that Chapel users simultaneously…
  …want things fixed / changed in the language
  …don’t like having their existing code break due to a new release

• **Chapel 2.0**: An upcoming release in which we…
  …commit to not breaking core language features
  …switch to semantic versioning

• Chapel 1.20 and 1.22 were the first two release candidates for Chapel 2.0
  • If we reach Chapel 1.23 without new breaking changes, it will be Chapel 2.0
Chapel 2.0: Progress since CHIUW 2019

• Short list of key remaining feature areas to focus on (from CHIUW 2019 slides):
  o initializers (replacement for constructors)
  o memory managed classes / lifetime checking
  o nilable vs. non-nilable class types
  o UTF-8 strings
  o move collection APIs from arrays/domains to standard types in the library
  o constrained generics (?)
Chapel 2.0: Progress since CHIUW 2019

- Short list of key remaining feature areas to focus on (from CHIUW 2019 slides):
  - initializers (replacement for constructors)
  - memory managed classes / lifetime checking
  - nilable vs. non-nilable class types
  - UTF-8 strings
  - move collection APIs from arrays/domains to standard types in the library
    - constrained generics (?)
Chapel 2.0: Progress since CHIUW 2019

• Short list of key remaining feature areas to focus on (from CHIUW 2019 slides):
  ✓ initializers (replacement for constructors)
  ✓ memory managed classes / lifetime checking
  ✓ nilable vs. non-nilable class types
  ✓ UTF-8 strings
  ✓ move collection APIs from arrays/domains to standard types in the library
    o constrained generics (?)

• Plus:
  ✓ module / namespace improvements
  ✓ partial instantiation
  ✓ 0- vs. 1-based indexing changes
  ✓ split-initialization, copy elision, deinitialization
  ✓ ‘bytes’ type
  ✓ and more…
Chapel 2.0: Status

• We think that Chapel 1.22 is a strong candidate for Chapel 2.0
  • However, that decision is ultimately up to users and devs like yourselves
  • Note that things can still evolve after Chapel 2.0, simply not in breaking ways

• The next pair of talks will cover more on this topic:
  • Michael Ferguson’s on stability in Chapel
  • Lydia Duncan’s on recent improvements for modules

• Also see Chapel’s release notes which cover language changes extensively
Chapel Performance Improvements
Perf Highlights (1.20): Unordered Optimization

- Enabled unordered optimization by default—improved existing codes w/ no edits

```plaintext
forall (_, r) in zip(Updates, RAStream()) do
    T[r & indexMask].xor(r);
```

![RA Performance (GUPS)](chart)

- Chapel 1.20
- Chapel 1.19
- MPI (bucketing)
Perf Highlights (1.20): Block Bulk Transfer

• Tuned and enabled bulk transfers on block-distributed arrays by default

\[ \text{locArr}[2..n-1] = \text{MyBlockArr}[lo..#(n-2)]; \]
Perf Highlights (1.21): Distributed Array Creation

• Reduced overheads associated with creating distributed domains and arrays

```chapel
var D = {1..n} dmapped Block({1..n}),
A: [D] real;
```

• At 512 locales: 90x fewer GETs, 15x faster
Perf Highlights (1.21): Arkouda Optimizations

- See Elliot Ronaghan’s talk later this morning
Python Interoperability
Python→Chapel Interoperability

```chapel
// foo.chpl
export proc hello() {
    coforall loc in Locales do on loc {
        writeln("Hello from locale ", loc);
    }
}
```

```python
# use_foo.py
import foo
# specify number of locales
foo.chpl_setup(2)
foo.hello()
```

```
$ chpl --library-python --comm=gasnet foo.chpl
$ python3 use_foo.py
Hello from locale 1
Hello from locale 0
```
Additional Highlights since CHIUW 2019
Notable Uses of Chapel

see Keynote and Applications of Chapel talks this afternoon

ChplUltra: Simulating Ultralight Dark Matter
Nikhil Padmanabhan et al.
Yale University

Arkouda open-sourced October 2019

CHGL: Chapel Hypergraph Library
Jesun Firoz, Cliff Joslyn, et al.
PNNL

Arkouda: NumPy at Massive Scale
Mike Merrill, Bill Reus, et al.
US DOD

CHAMPS: 3D Computational Fluid Dynamics
Simon Bourgault-Côté, Matthieu Parenteau, et al.
École Polytechnique Montréal

ChOp: Chapel-based Optimization
Tiago Carneiro, Nouredine Melab, et al.
INRIA Lille, France

CrayAI: Distributed Machine Learning
Cray, a Hewlett Packard Enterprise Company

For more information, see: https://chapel-lang.org/poweredby.html
Chapel odometer flips since CHIUW 2019

• 10,000 : number of tests in the full Chapel test suite (currently ~11,640)
• 1,000  : number of stars on GitHub (currently 1,081)
• 100    : number of individual GitHub contributors (currently 106)
Contributors to Chapel 1.20–1.22 releases

- Ben Albrecht, Cray/HPE
- Souris Ash, individual contributor
- Himanshu Bhandari, individual contributor
- Akshansh Bhanjana, individual contributor
- Ankush Bhardwaj, GSoC 2020 student from Royal Global University
- Paul Cassella, Cray/HPE
- Brad Chamberlain, Cray/HPE
- Sanket Chaudhari, individual contributor
- Soohoon Choi, Cray/HPE
- Anway De, individual contributor
- Garvit Dewan, GSoC 2019 student from IIT Roorkee
- Krishna Kumar Dey, GSoC 2019 student from Indian Institute of Information Technology, Sri City (former individual contributor)
- Lydia Duncan, Cray/HPE
- Omar Elawady, individual contributor
- Michael Ferguson, Cray/HPE
- Rahul Ghangas, GSoC 2020 student from Australian National University
- Ben Harshbarger, Cray/HPE
- Samuel Howard, individual contributor
- Sai Rajendra Immadi, individual contributor
- David Iten, Cray/HPE

- Louis Jenkins, Pacific Northwest National Laboratory, GSoC 2019 mentor (former individual contributor, GSoC 2017 student from Bloomsburg University)
- Engin Kayraklioglu, Cray/HPE (former GSoC 2017 mentor, Cray Inc. intern from George Washington University)
- David Keaton, Cray Inc.
- Sriram Srinivasan Krishna, individual contributor
- Kshitiz Lakhotia, individual contributor
- Vassily Litvinov, Cray/HPE
- David Longnecker, Cray/HPE
- Aniket Mathur, GSoC 2020 student from IIT Roorkee
- Mohammed Nafees, GSoC 2019 student from University of Waterloo
- Nikhil Padmanabhan, Yale University
- Owen Plambeck, Cray Inc. intern from Haverford College
- Audrey Pratt, Cray/HPE
- Yujia Qiao, GSoC 2020 student from Huazhong University of Science and Technology
- Elliot Ronaghan, Cray/HPE
- Raj Shekhar, individual contributor
- Jenna Hoole Starkey, Cray/HPE
- Greg Titus, Cray/HPE
- David Tjong, individual contributor
- Anubhav Ujjawal, individual contributor
- Karlon West, Cray/HPE
- Alvis Wong, GSoC 2019 student from University of Waterloo
What’s Next?
Next Steps

• **Obvious Continuations:**
  - Chapel 2.0
  - User Support
  - Performance
  - Portability, especially to Shasta
Next Steps

• **New / Renewed Efforts:**
  • **GPUs:** add support (see “Chapel on GPUs” session this afternoon)
  • **LLVM back-end:** enable by default, improve vectorization
  • **Compiler:** improvements to organization, speed, and capabilities
  • **Standard modules:** review and stabilize interfaces
Next Steps

• **Community-related Efforts:**
  
  • **Mailing Lists:** retire SourceForge mailing lists in favor of Discourse
  
  • **Blog:** launch a Chapel blog series
  
  • **Website:** revamp and modernize
The 3rd Annual Parallel Applications Workshop, Alternatives To MPI+X

Monday, November 16th, 2020

Held in conjunction with SC20

Submission deadline July 24
(like CHIUW, accepts papers and talks)

https://sourceryinstitute.github.io/PAW/
The fifth edition of ESPM2 workshop, being proposed to be held as a full-day meeting with the Supercomputing (SC2020) conference in Atlanta, Georgia focuses on programming models and runtimes for extreme scale systems. Next generation architectures and systems being deployed are characterized by high concurrency, low memory per-core, and multiple levels of hierarchy and heterogeneity. These characteristics bring out new challenges in energy efficiency, fault-tolerance and scalability. It is commonly believed that software has the biggest share of the responsibility to tackle these challenges. In other words, this responsibility is delegated to the next generation programming models and their associated middleware/runtimes. This workshop focuses on different aspects of programming models such as task-based parallelism (Charm++, OCR, Habanero, Legion, X10, HPX, etc.), PIGAS (OpenSHMEM, UPC, CAF, Chapel, UPC++, etc.), BigData (Hadoop, Spark, Dask etc), Machine Learning (NVIDIA RAPIDS, Scikit-learn etc.), Deep Learning (Caffe, Microsoft CNTK, Google TensorFlow, Facebook PyTorch), directive-based languages (OpenMP, OpenACC) and Hybrid MPI+X. It also focuses on their associated middleware (unified runtimes, interoperability for hybrid programming, tight integration of MPI+X, and support for accelerators/FPGAs) for next generation systems and architectures.

The ultimate objective of the ESPM2 workshop is to serve as a forum that brings together researchers from academia and industry working in the areas of programming models, runtime systems, compilation and languages, and application developers.

ESPM2 2020 will be held as a full day workshop in conjunction with the SuperComputing (SC 2020), on Sunday November 15th in Atlanta, Georgia, USA.

Topics
ESPM2 2020 welcomes original submissions in a range of areas, including but not limited to:
- New programming models, languages and constructs for exploiting high concurrency and heterogeneity
- Experience with and improvements for existing parallel languages and run-time environments such as:
  - MPI
  - PIGAS (OpenSHMEM, UPC, CAF, Chapel, UPC++, etc.)
  - Directive-based programming (OpenMP, OpenACC)
  - Asynchronous Task-based models (Charm++, OCR, Habanero,
  - Legion, X10, HPX, etc)
  - Hybrid MPI+X models
  - BigData (Hadoop, Spark, Dask etc.)
- Machine Learning (NVIDIA RAPIDS, Scikit-learn etc.), and
- Deep Learning (Microsoft CNTK, TensorFlow, PyTorch)
- Parallel compilers, programming tools, and environments
- Programming environments for heterogeneous multi-core systems and accelerators such as KNL, OpenPOWER, ARM, GPUs, FPGAs, MICs, and DSPs

Submission deadline August 31
(full research papers only, I believe)

http://nowlab.cse.ohio-state.edu/espm2/
Chapel may be in quarantine, but we are hiring
Summary

• A lot of great progress has been made over the past year

• The community seems strong and growing in terms of users and developers

• We’re happy to have the excuse to hold CHIUW online this year
  • And, we’re glad you’re all here
Welcome to CHIUW 2020!
The 7th Annual Chapel Implementers and Users Workshop

May 22, 2020

chapel_info@cray.com
chapel-lang.org
@ChapelLanguage
CHIUW 2020: Technical Details

• Ask questions to the speakers using Zoom’s Q&A tool
  • Session chairs will help relay the questions to speakers

• Use the chat window for technical help / questions
  • Or to chat socially during breaks and pauses

• Please use professionalism and common sense in all cases

pre: Chapel 101 (optional)

now: Welcome, State of the Project

9:15: talks: Chapel Language Evolution

10:20: talks: Arkouda, including our keynote talk (next slide)

12:00: talks: Applications of Chapel

1:55: talks: Chapel on GPUs

2:50: talks: Implementing Chapel

3:35: Open Discussion Session

future: Online Coding Day?
We’ll be right back!

CHIUW 2020
May 22, 2020

chapel_info@cray.com
chapel-lang.org
@ChapelLanguage
SAFE HARBOR STATEMENT

This presentation may contain forward-looking statements that are based on our current expectations. Forward looking statements may include statements about our financial guidance and expected operating results, our opportunities and future potential, our product development and new product introduction plans, our ability to expand and penetrate our addressable markets and other statements that are not historical facts.

These statements are only predictions and actual results may materially vary from those projected. Please refer to Cray's documents filed with the SEC from time to time concerning factors that could affect the Company and these forward-looking statements.
THANK YOU

QUESTIONS?

bradc@cray.com
@ChapelLanguage
chapel-lang.org

cray.com
@cray_inc
linkedin.com/company/cray-inc/