
Hewlett Packard
Enterprise
STATE OF
THE CHAPEL PROJECT
Brad Chamberlain
June 4, 2021

WHAT IS CHAPEL?

Chapel: A modern parallel programming language

- portable & scalable
- open-source & collaborative

Goals:

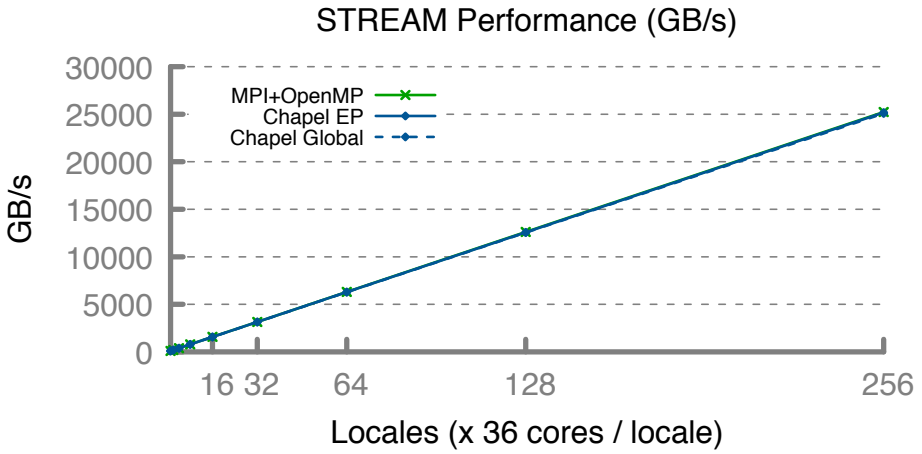
- Support general parallel programming
- Make parallel programming at scale far more productive

CHAPEL BENCHMARKS TEND TO BE CONCISE, CLEAR, AND COMPETITIVE

STREAM TRIAD: C + MPI + OPENMP

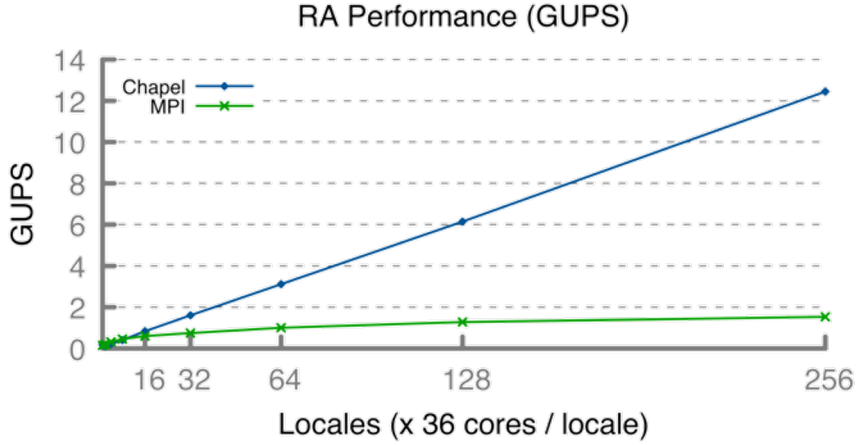
```
#include <hpcc.h>
#ifdef _OPENMP
#include <omp.h>
#endif

static int VectorSize;
static double *a, *b, *c;

int HPCC_StarStream(HPCC_Params *params) {
  int myRank, commSize;
  int rv, errCount;
  MPI_Comm comm = MPI_COMM_WORLD;

  MPI_Comm_size( comm, &commSize );
  MPI_Comm_rank( comm, &myRank );

  rv = HPCC_Stream( params, 0 == myRank);
  MPI_Reduce( &rv, &errCount, 1, MPI_INT, MPI_SUM, 0, comm );

  return errCount;
}

int HPCC_Stream(HPCC_Params *params, int doIO) {
  register int j;
  double  scalar;

  VectorSize = HPCC_LocalVectorSize( params, 3, sizeof(double), 0 );

  a = HPCC_XMALLOC( double, VectorSize );
  b = HPCC_XMALLOC( double, VectorSize );
  c = HPCC_XMALLOC( double, VectorSize );

  if (!a || !b || !c) {
    if (c) HPCC_free(c);
    if (b) HPCC_free(b);
    if (a) HPCC_free(a);
    if (doIO) {
      fprintf( outFile, "Failed to allocate memory
      fclose( outFile );
    }
    return 1;
  }

#ifdef _OPENMP
#pragma omp parallel for
#endif
  for (j=0; j<VectorSize; j++) {
    b[j] = 2.0;
    c[j] = 1.0;
  }
  scalar = 3.0;

#ifdef _OPENMP
#pragma omp parallel for
#endif
  for (j=0; j<VectorSize; j++)
    a[j] = b[j]+scalar*c[j];

  HPCC_free(c);
  HPCC_free(b);
  HPCC_free(a);

  return 0;
}
```

```
use BlockDist;

config const m = 1000,
             alpha = 3.0;
const Dom = {1..m} dmapped …;
var A, B, C: [Dom] real;

B = 2.0;
C = 1.0;

A = B + alpha * C;
```

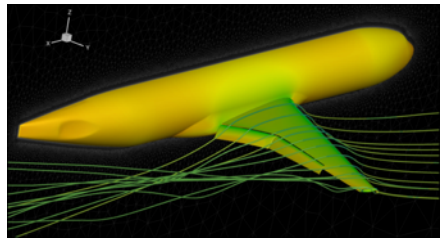


HPCC RA: MPI KERNEL

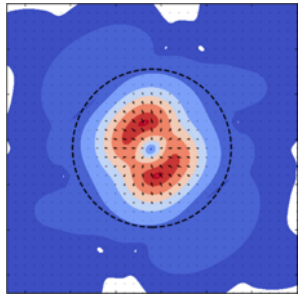
```
…
forall (_, r) in zip(Updates, RAStream()) do
  T[r & indexMask].xor(r);
…
```



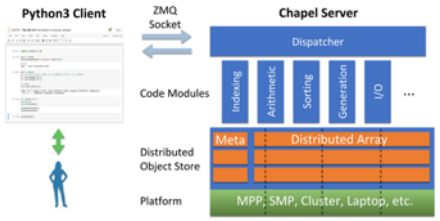
NOTABLE APPLICATIONS OF CHAPEL



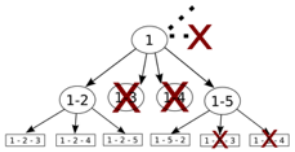
CHAMPS: 3D Comp. Fluid Dynamics
Éric Laurendeau, Simon Bourgault-Côté, Matthieu Parenteau, Anthony Bouchard, Hélène Papillon Laroche, et al.
École Polytechnique Montréal



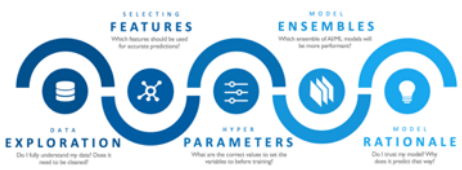
ChplUltra: Simulating Ultralight Dark Matter
Nikhil Padmanabhan, J. Luna Zagorac, Richard Easther, *et al.*
Yale University / University of Auckland



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



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

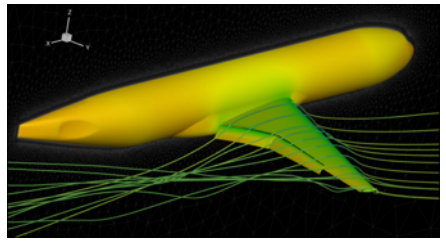


CrayAI: Distributed Machine Learning
Hewlett Packard Enterprise

?

Your Project Here?

NOTABLE APPLICATIONS OF CHAPEL

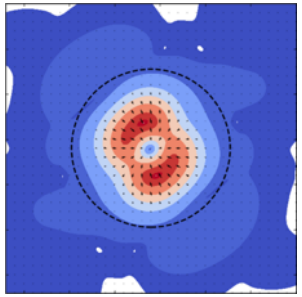


CHAMPS: 3D Comp. Fluid Dynamics

Éric Laurendeau, Simon Bourgault-Côté, Matthieu Parenteau, Anthony Bouchard, Hélène Papillon Laroche, et al.

École Polytechnique Montréal

Keynote by Éric @ 10am
Technical Talks by Hélène and Anthony @ 11:15 and 2pm

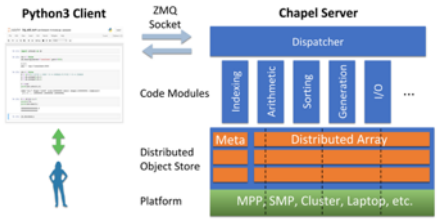


ChplUltra: Simulating Ultralight Dark Matter

Nikhil Padmanabhan, J. Luna Zagorac, Richard Easther, *et al.*

Yale University / University of Auckland

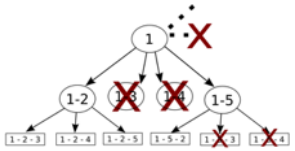
Lightning Talk by Nikhil @ 2:45
CHIUW 2020 Technical Talk by Nikhil online



Arkouda: NumPy at Massive Scale

Mike Merrill, Bill Reus, et al.

US DOD



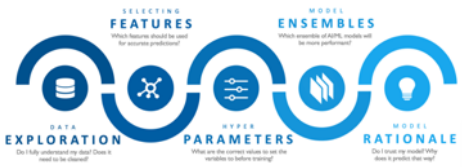
ChOp: Chapel-based Optimization

Tiago Carneiro, Nouredine Melab, *et al.*

INRIA Lille, France

Technical Talk by Tiago @ 11:35

Technical Talks by Ben McDonald and Zhihui Du @ 12:45 and 1:40
CHIUW 2020 Keynote by Bill online



CrayAI: Distributed Machine Learning

Hewlett Packard Enterprise

Technical Talk by Ben Albrecht @ 2:15

?

Your Project Here?

Technical Talk by Damian McGuckin @ 11:50

HIGHLIGHTS* FROM THE PAST YEAR IN THE LIFE OF…

(* = that I'm aware of anyway…)

…Arkouda:

- talk at SciPy 2020: *Arkouda: Terascale Data Science at Interactive Rates*
- Arkouda Hack-a-thon (archived at YouTube)
- weekly Arkouda Zoom call to discuss algorithms, code, and methods

…ChOp:

- HPCS 2020 paper: *Towards Chapel-based Exascale Tree Search Algorithms: dealing with multiple GPU accelerators*
  - won the HPCS 2020 **Outstanding Paper Award**
  - in collaboration with Georgia Tech
- Swarm and Evolutionary Computation journal article: *A Comparative Study of High-Productivity High-Performance Programming Languages for Parallel Metaheuristics*

…CHAMPS:

- paper at AIAA SciTech 2021: *Development of Parallel CFD Applications with the Chapel Programming Language*
- also collaborating with Georgia Tech w.r.t. GPU programming
- exciting workshops on the horizon (more in today's talks)

(links to papers, slides, and videos available from Chapel's Papers and Publications and Presentations pages)

THE CHAPEL TEAM AT HPE IS GROWING

CHIUW 2020:

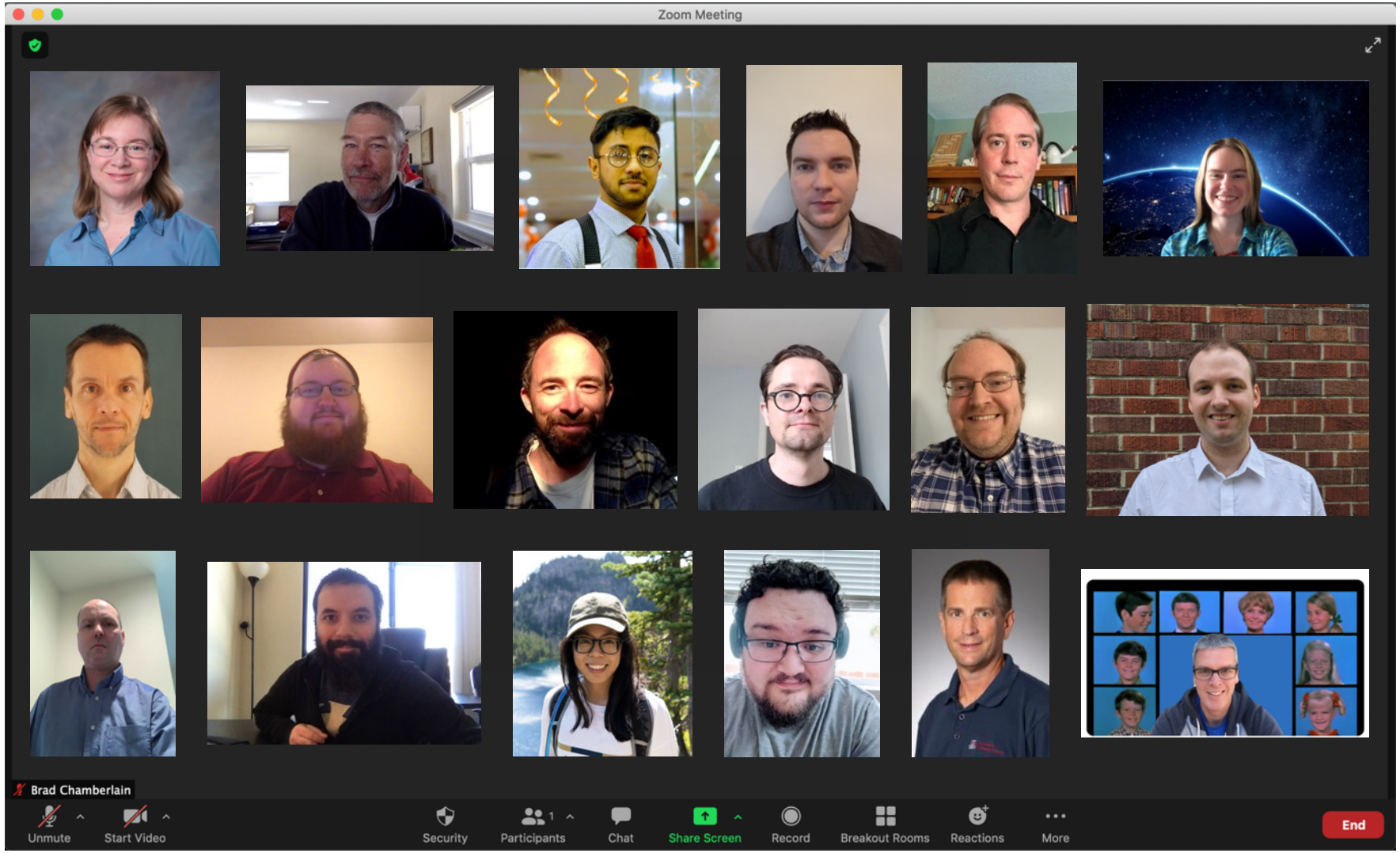
- 11 engineers
- 0.5 managers

CHIUW 2021:

- 15 engineers
  +2 more starting June–July
- 1.5 managers
- 1 summer intern

Goal: 19 FTEs by Dec 2021

- 1 open position currently
- 1 more to come
- chapel-lang.org/jobs.html



INTRODUCING OUR NEW CHAPEL MANAGER AT HPE

Michelle Mills Strout

- Professor at University of Arizona
  - Previously:
    - Sabbatical at Australian National University
    - Visiting Professor at Waseda University
    - Faculty member at Colorado State University
    - Postdoc at Argonne National Laboratory
    - PhD, MS, BS from UCSD
- Focus on compilers and HPC
  - Upcoming invited talk at PLDI 2021
- Has a strong history with Chapel / CHIUW
  - ICS 2015 paper on Diamond Tiling in Chapel
  - gave talk on Diamond Tiling at CHIUW 2015
  - co-author on a CHIUW 2018 paper on imperfectly nested loops
  - lightning talk at CHIUW 2019
  - PC member at CHIUW 2015–2018
  - Session chair at CHIUW 2015, 2019

OUTLINE

- ✓Chapel Context, Users, and Team
- Recent Chapel Releases
  - Programming Improvements
  - Performance Improvements
- Outreach / Community Highlights
- Wrap-up


RECENT CHAPEL RELEASES

CHAPEL RELEASES SINCE CHIUW 2020

Chapel 1.23.0 (October 15, 2020), focused on:

- language stability
- performance optimizations
- collection types

Chapel 1.24.0 (March 18, 2021), focused on:

- language stability
- performance optimizations
- LLVM back-end readiness

Chapel 1.24.1 (April 15, 2021): an update release, motivated by:

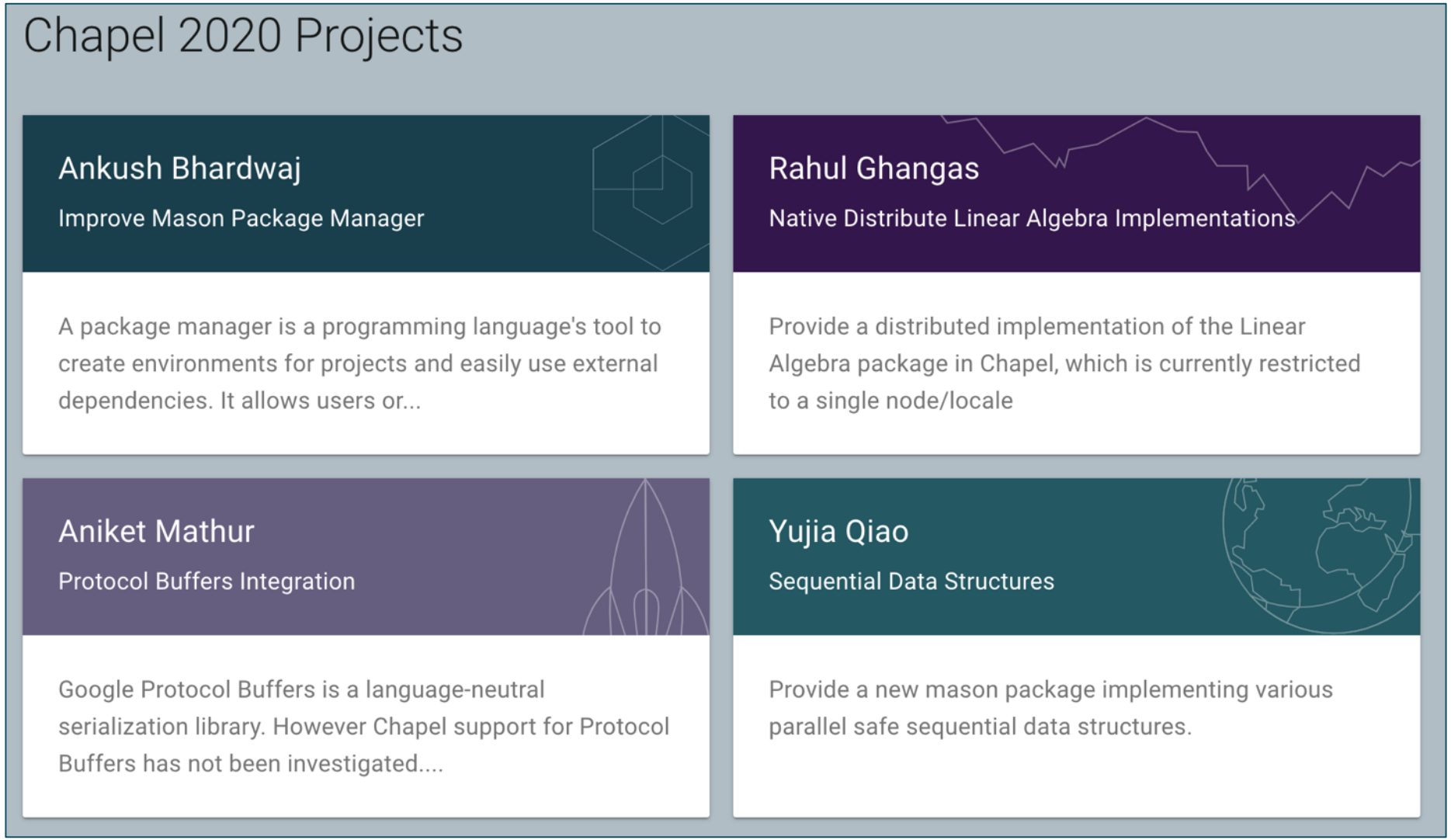
- Infiniband performance improvements
- Portability fixes for HPE Cray EX

CONTRIBUTORS TO CHAPEL 1.23–1.24

- Ben Albrecht, HPE
- Ankush Bhardwaj, GSoC 2020 student from Royal Global University
- Paul Cassella, HPE
- Brad Chamberlain, HPE
- R Chinmay, individual contributor
- Soohoon Choi, HPE
- Cristian-loan Condruz, individual contributor
- Garvit Dewan, GSoC 2020 mentor, GSoC 2019 student from Indian Institute of Technology Roorkee
- Krishna Kumar Dey, GSoC 2020 mentor, GSoC 2019 student from Indian Institute of Information Technology, Sri City
- Nelson Luís Dias, individual contributor
- Lydia Duncan, HPE
- Prashanth Duvvuri, individual contributor
- Michael Ferguson, HPE
- Rahul Ghangas, GSoC 2020 student from Australian National University
- Piyush Gupta, individual contributor
- Ben Harshbarger, HPE
- Sai Rajendra Immadi, individual contributor
- David Iten, HPE
- Engin Kayraklioglu, HPE (former GSoC 2017 mentor, Cray Inc. intern from George Washington University)
- Lee Killough, HPE
- Vassily Litvinov, HPE
- Priyank Lohariwal, individual contributor
- David Longnecker, HPE
- Aniket Mathur, GSoC 2020 student from Indian Institute of Technology Roorkee
- Ben McDonald, HPE intern from Gonzaga University
- Erin Melia, individual contributor
- Ram Nad, individual contributor
- Divye Nayyar, individual contributor
- Sarah Nguyen, HPE
- Nikhil Padmanabhan, Yale University
- Parth Sarthi Prasad, individual contributor
- Yujia Qiao, GSoC 2020 student from Huazhong University of Science and Technology
- Elliot Ronaghan, HPE
- Mohammed Sharfuddin, individual contributor
- Raj Shekhar, individual contributor
- Jenna Hoole Starkey, HPE
- Michelle Mills Strout, HPE
- Greg Titus, HPE
- Joe Tursi, HPE
- Karlon West, HPE
- Souris Ash, individual contributor

HPE Employees | Individual Contributors | Google Summer of Code

GOOGLE SUMMER OF CODE 2020: FOUR SUCCESSFUL PROJECTS



GOOGLE SUMMER OF CODE 2021: JUST GETTING STARTED

See the GSoC website for more information


PROGRAMMING IMPROVEMENTS

LANGUAGE HIGHLIGHTS SINCE CHIUW 2020

- The biggest language highlight since CHIUW 2020…

  …is that there aren't any particularly impactful breaking language changes

- Contrast this with the previous few years:
  - **Chapel 1.17:** shifted from constructors to initializers
  - **Chapel 1.18:** switched classes to managed memory
  - **Chapel 1.19:** changed throw/catch to use 'owned' errors
  - **Chapel 1.20:** made classes non-nilable by default
  - **Chapel 1.21:** added support for split initialization and copy elision
  - **Chapel 1.22:** switched from 1-based implicit indices to 0-based

CHAPEL 2.0: CONCEPT AND STATUS

- **Chapel 2.0:** a forthcoming release in which core language features can be considered stable
  - to avoid breaking users' codes with each release
  - to rally potential users to give Chapel another look

- **Chapel 1.23–1.24:** language improvements in support of Chapel 2.0:
  - Namespace fixes
  - Implicit accesses to sync/single
  - Point-of-Instantiation fixes
  - Array initialization fixes
  - [De]Initialization order fixes
  - Refining type conversion features

- **What remains?**
  - User-defined collections, particularly storing non-nilable classes
  - Constrained generic interfaces
  - Standard library stabilization…

CHAPEL 2.0: STANDARD LIBRARIES

- Since CHIUW 2020, we've realized that Chapel 2.0 should also involve stabilizing key standard libraries
  - Some subset of:
    - Builtins, Chapel Environment Variables
    - Heap, List, Map, Set
    - CommDiagnostics, Memory
    - FileSystem, Automatic IO, IO, Path
    - Reflection, Types
    - BigInteger, BitOps, GMP, Math, Random
    - Barriers, DynamicIters, VectorizingIterator
    - CPtr, Spawn, Sys, SysBasic, SysCTypes, SysError
    - DateTime, Help, Regexp, Time, Version
    - HaltWrappers
  - As well as library-like features in the language, such as methods and functions on standard types:
    - String, Bytes
    - Ranges, Domains, Arrays
    - Shared, Owned

CHAPEL 2.0: STANDARD LIBRARIES

- Since CHIUW 2020, we've realized that Chapel 2.0 should also involve stabilizing key standard libraries
  - Some subset of:
    - **Builtins**, Chapel Environment Variables
    - Heap, List, Map, Set
    - **CommDiagnostics**, Memory
    - FileSystem, Automatic IO, IO, **Path**
    - **Reflection**, Types
    - **BigInteger**, BitOps, GMP, Math, Random
    - **Barriers**, DynamicIters, VectorizingIterator
    - CPtr, **Spawn**, **Sys**, SysBasic, SysCTypes, SysError
    - DateTime, Help, **Regexp**, **Time**, Version
    - HaltWrappers
  - As well as library-like features in the language, such as methods and functions on standard types:
    - String, Bytes
    - **Ranges**, Domains, Arrays
    - **Shared**, Owned

bold = has received a round of review so far

PERFORMANCE IMPROVEMENTS

PERFORMANCE OPTIMIZATIONS: HIGHLIGHTS SINCE CHIUW 2020

Compiler Optimizations:

- Automatic Local Access Optimization
- Automatic Aggregation Optimization

Runtime-based Optimizations:

- Remote Caching Improvements

Array Improvements:

- Parallel Array Initialization and Assignment
- Array / Domain Tracking Optimizations
- Array Swap Optimization
- Optimized Associative Domains / Arrays / Types
- Scan Optimizations

Portability

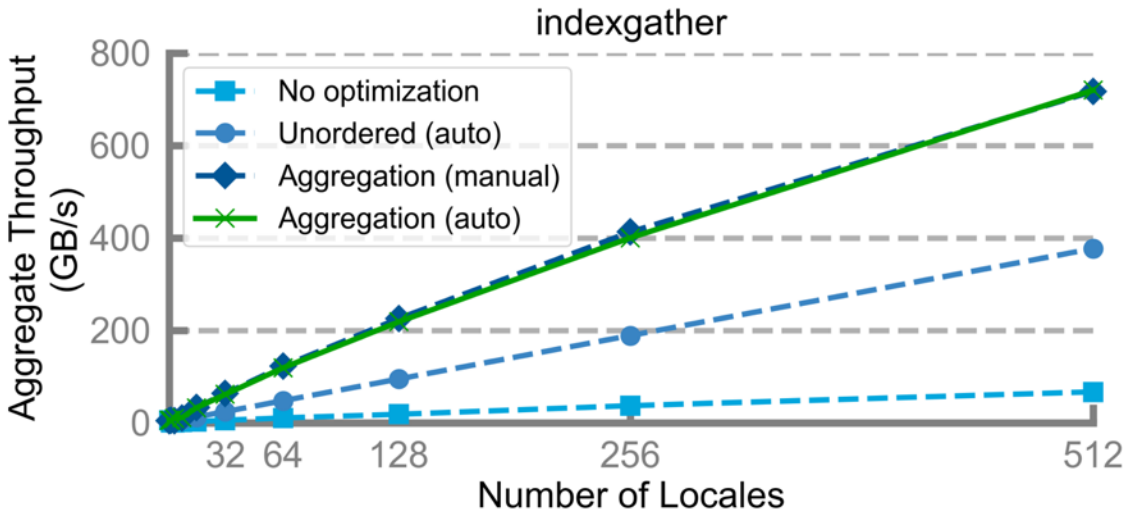
- Optimized Performance for InfiniBand systems

Memory Leak Improvements

Compilation Time Improvements

TWO REPRESENTATIVE PERFORMANCE GRAPHS

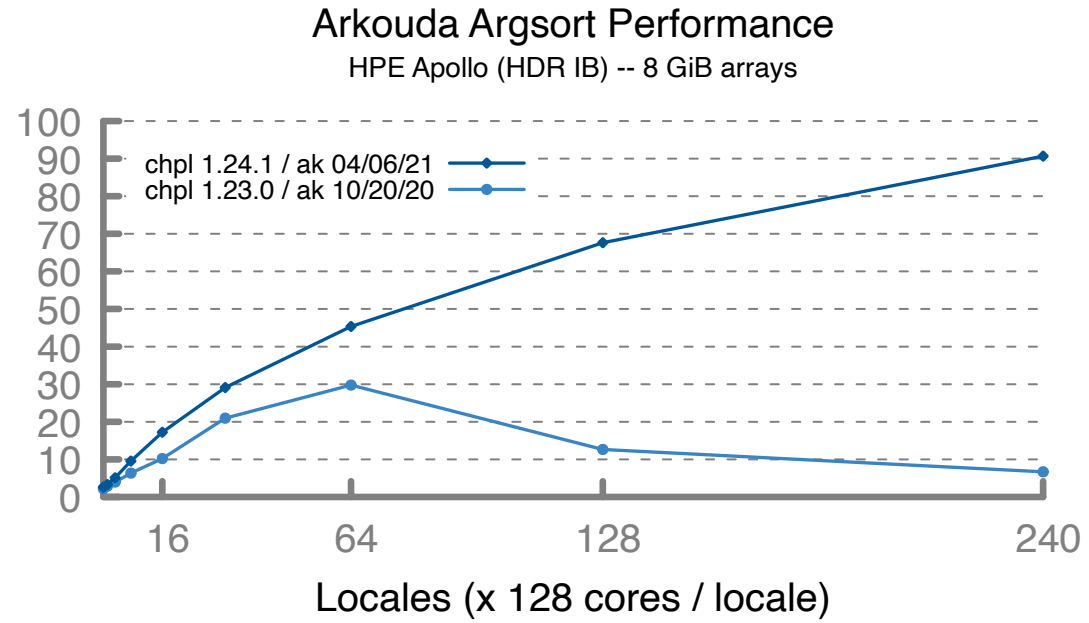
Compiler-Driven Optimization



- Chapel was designed for compiler analysis & opt.
  - Yet we've only recently started leveraging that

Talk by Engin Kayraklioglu @ 9:25

Portability-Focused Optimization



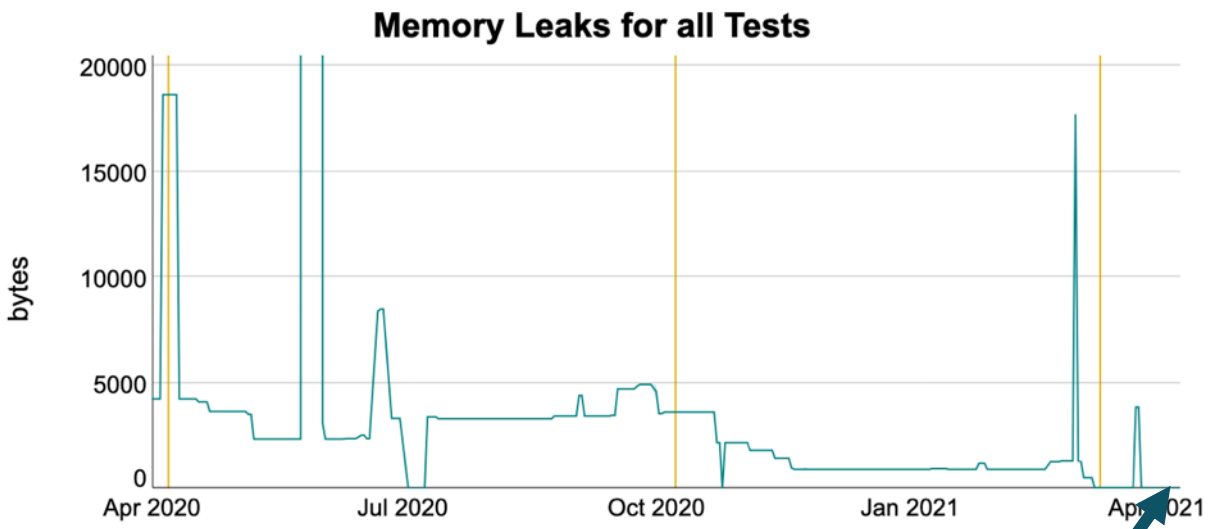
- Chapel was designed for portability
  - Yet we've historically focused primarily on Crays

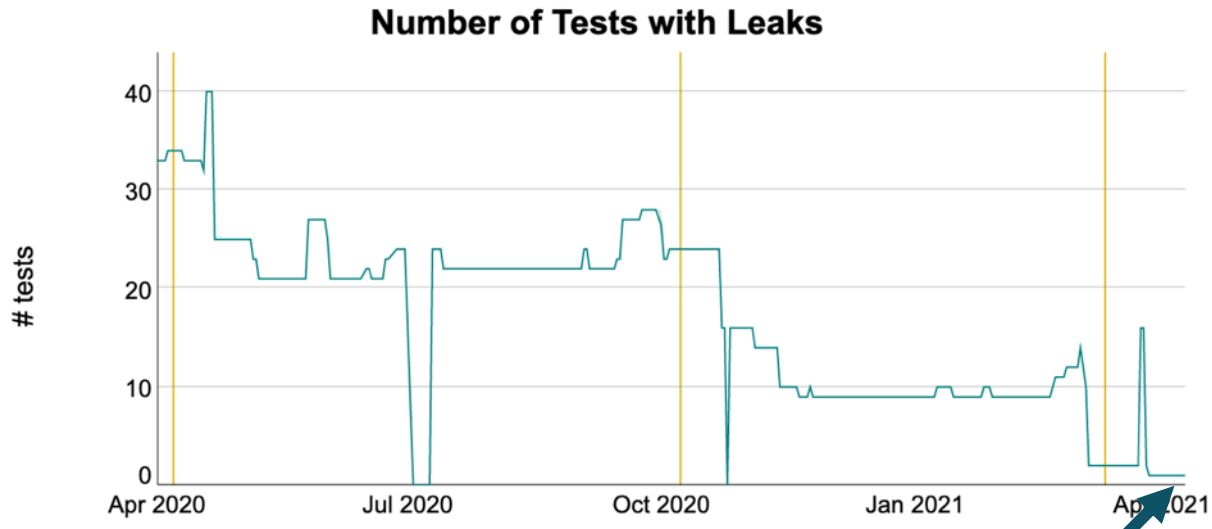
Talk by Elliot Ronaghan @ 9:05

Also, a Performance-focused Talk by Thomas Rolinger @ 1:05

MEMORY LEAKS: SINCE CHIUW 2020

- All known memory leaks have been closed





~18 KB leaked in 1.22

8 B leaked by design in 1.24.1

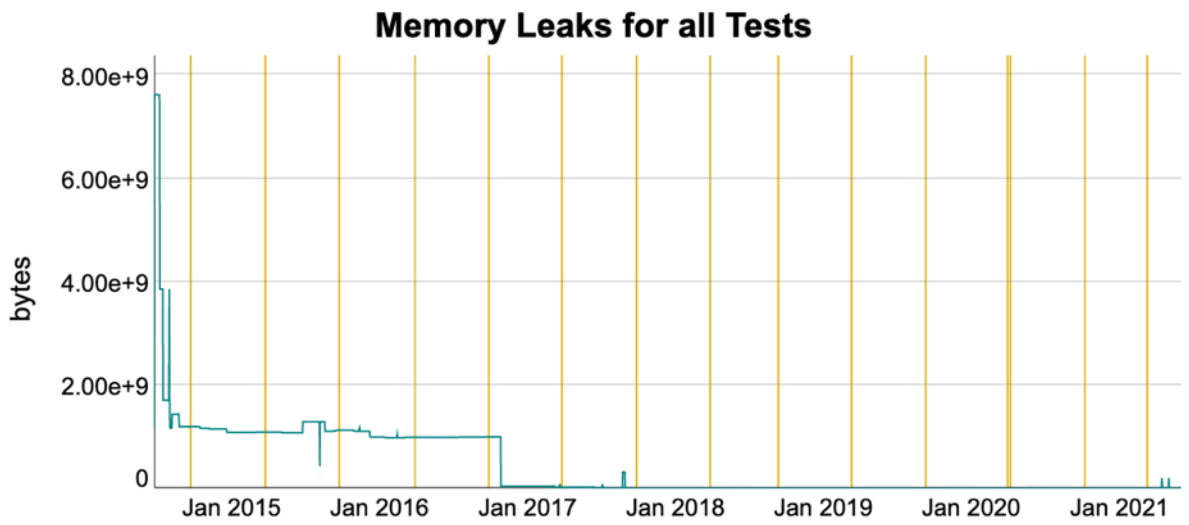
34 tests leaked in 1.22

1 test leaks by design in 1.24.1

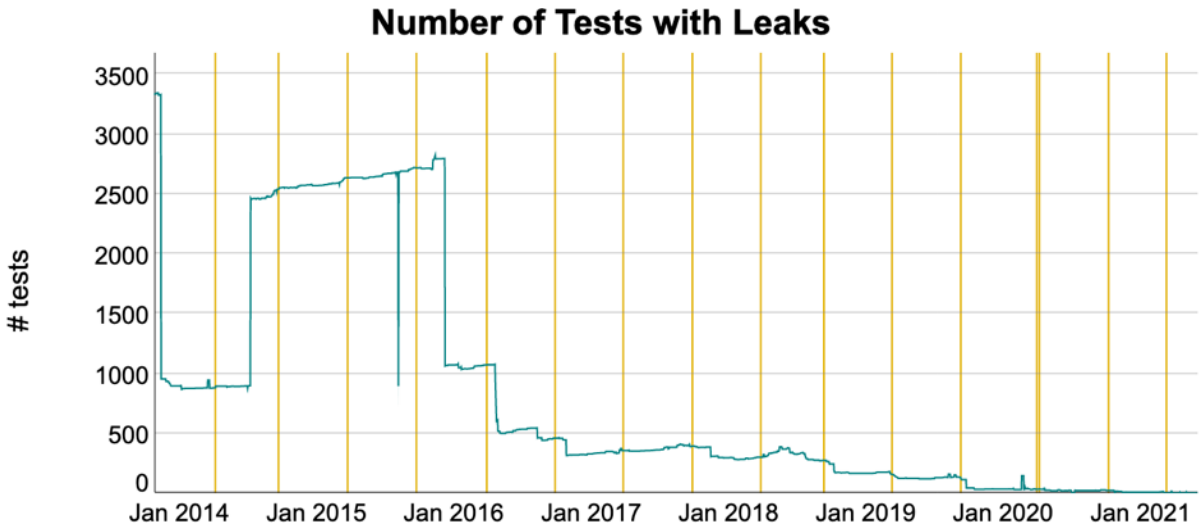
Nightly performance graphs available at: https://chapel-lang.org/perf-nightly.html

MEMORY LEAKS: SINCE CHIUW'S INCEPTION

- We've come a long way since CHIUW began…



7+ GB leaked in Chapel 1.8

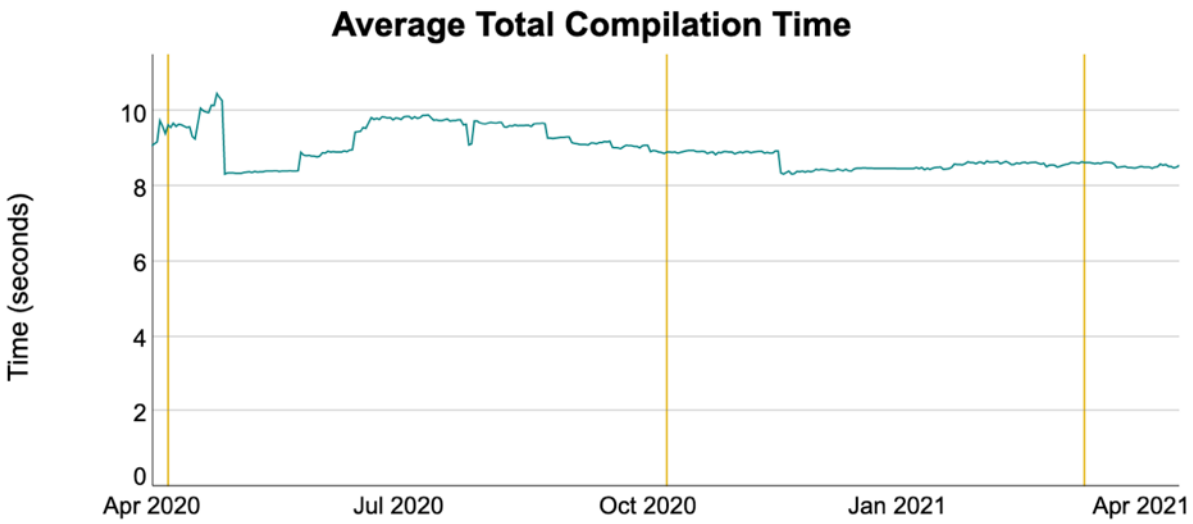


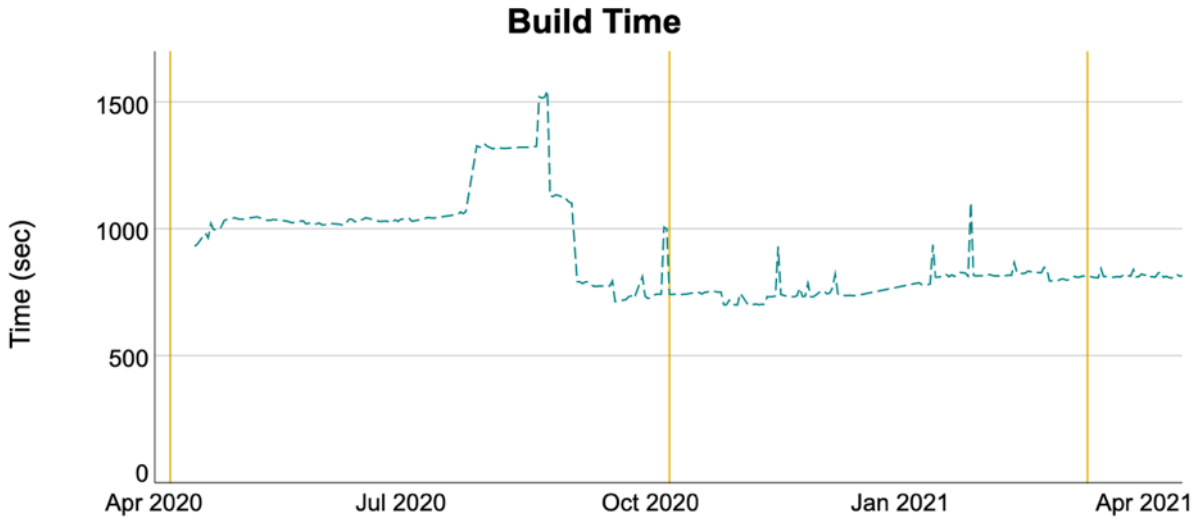
3300+ tests leaked in Chapel 1.8

Nightly performance graphs available at: https://chapel-lang.org/perf-nightly.html

COMPILATION TIME IMPROVEMENTS SINCE CHIUW 2020

- Compilation time has also generally improved
  - But often only modestly
  - Bigger improvements are warranted and desired





Nightly performance graphs available at: https://chapel-lang.org/perf-nightly.html

TWO KEY COMPILER INITIATIVES STARTED SINCE CHIUW 2021

1. A major compiler overhaul

- Improve compilation speeds and scalability
- Support interactive programming and tools
- Simplify learning curve for developers

Talk by Michael Ferguson @ 8:45

2. Support for GPU programming in Chapel

- extend Chapel's "any parallel algorithm on any parallel hardware" goal to include GPUs

```
on here.GPUSublocale() do
  forall (a, b, c) in zip(A, B, C) do
    a = b + alpha*c;
```

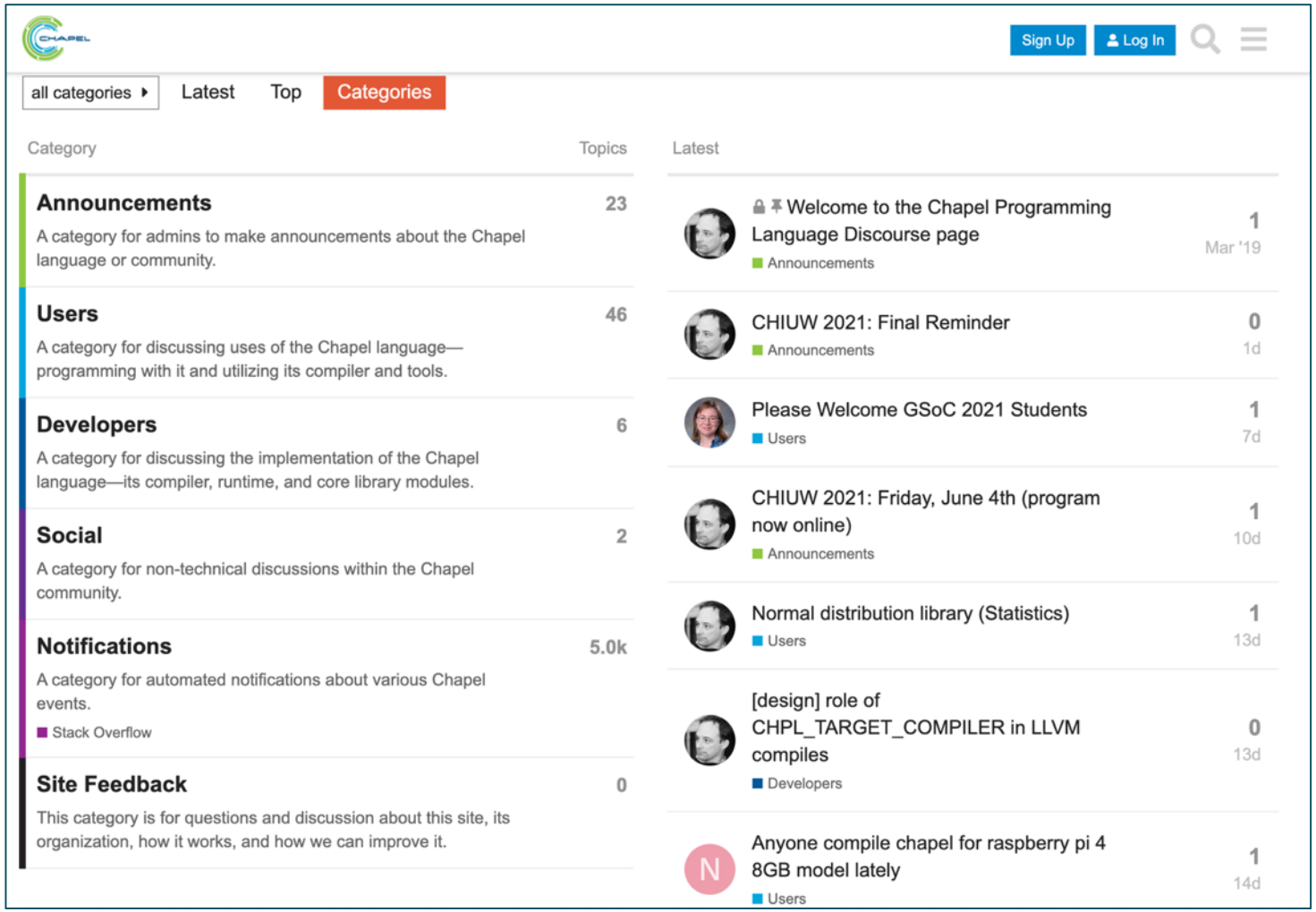
No talk by our group on this effort today…
Refer to the 1.24 release notes for an early look

GPU-related talks by Tiago, Akihiro Hayashi, and Anthony at 11:35, 12:25, 2pm

OUTREACH / COMMUNITY HIGHLIGHTS

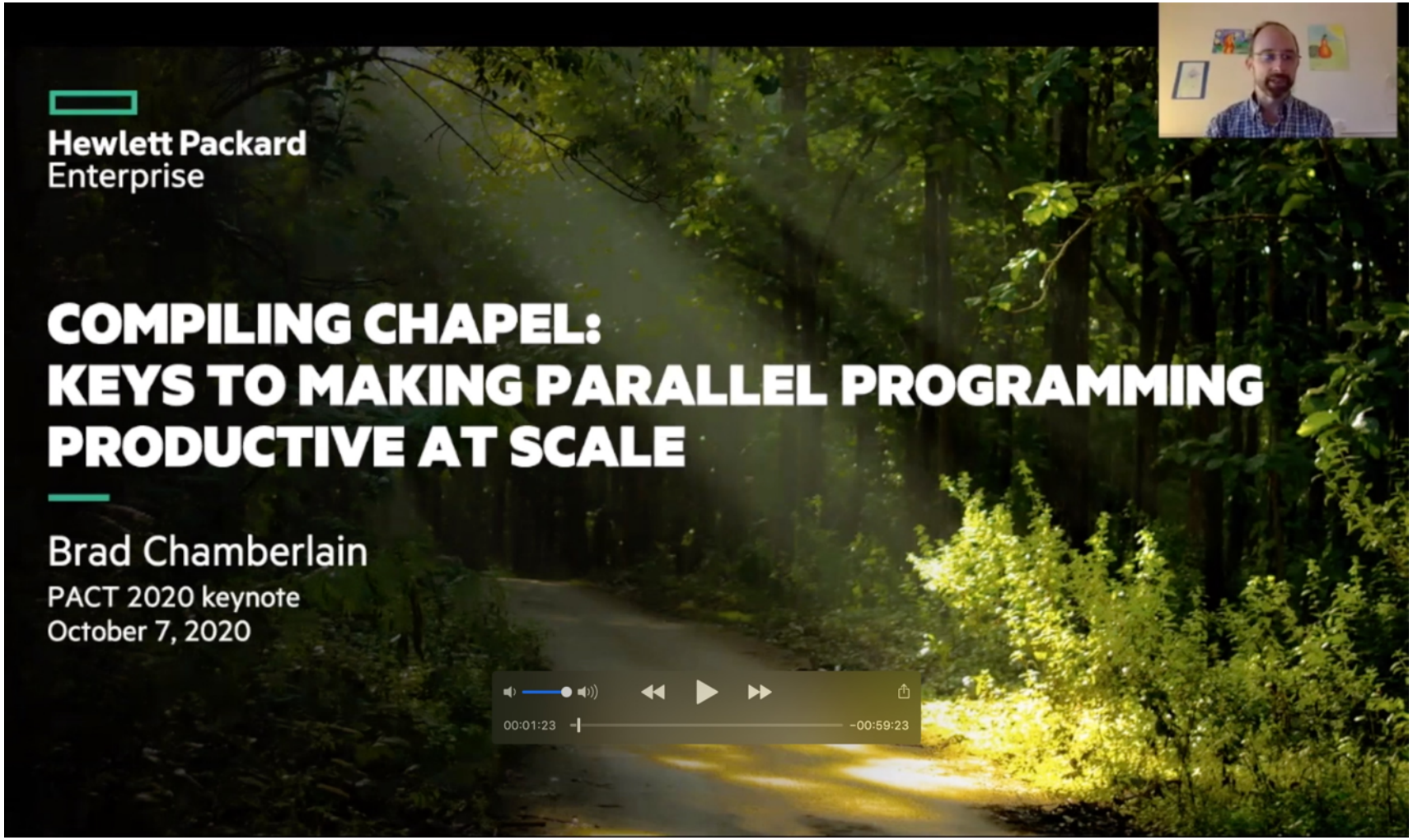
OUTREACH / COMMUNITY HIGHLIGHTS

- Sep: **Launched Discourse site**
  - Replaces SourceForge mailing lists
    - accessible as a web forum
    - or in mailing list mode
  - Complementary to:
    - Gitter: real-time chat
    - Stack Overflow: persistent Q&A
    - GitHub Issues: bugs, feature requests, …



OUTREACH / COMMUNITY HIGHLIGHTS

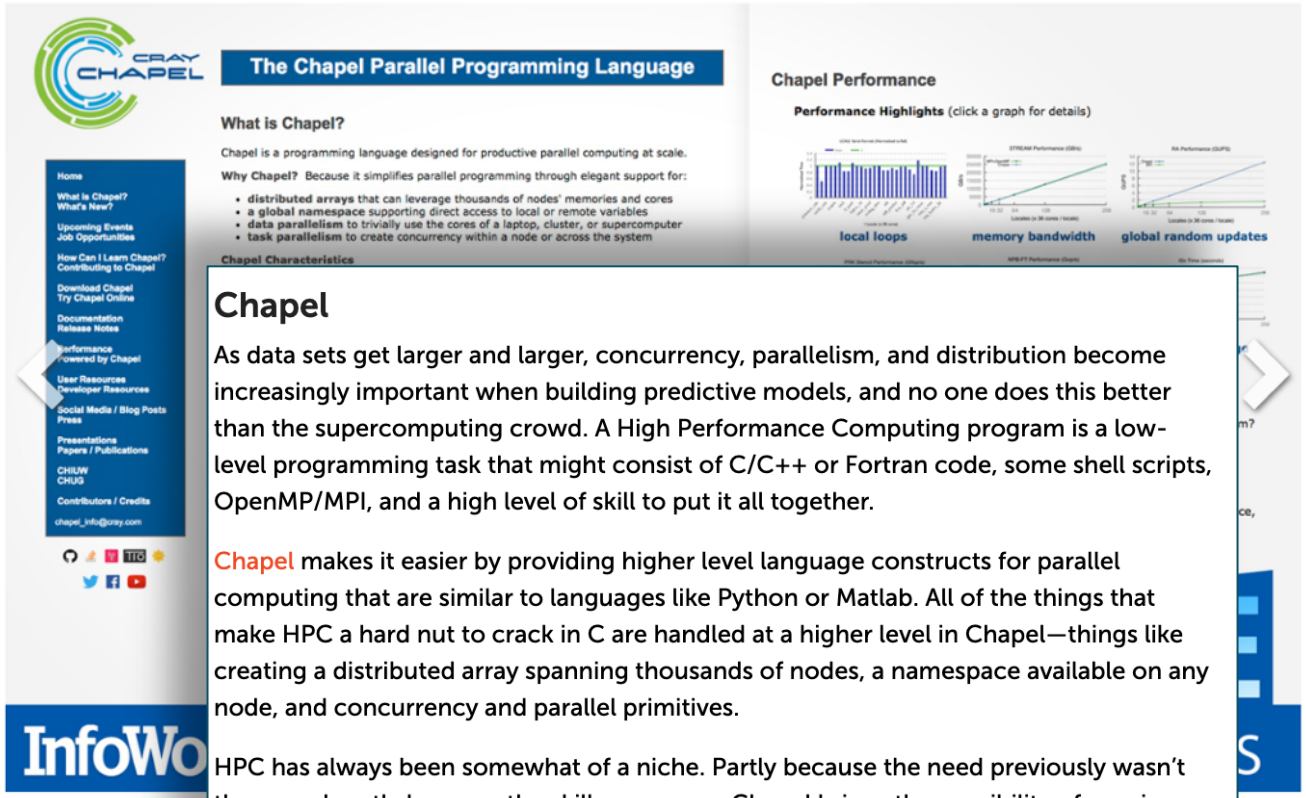
- Oct: **PACT'20 keynote**
  - topic: compiling Chapel
    - transformations
    - optimizations
    - [video | slides]



OUTREACH / COMMUNITY HIGHLIGHTS

- Oct: **2020 Bossie Award**
  - named one of 25 best open-source projects
  - others included:
    - Apache Arrow
    - Drupal
    - Jekyll
    - Redis
    - …

Chapel



Chapel

As data sets get larger and larger, concurrency, parallelism, and distribution become increasingly important when building predictive models, and no one does this better than the supercomputing crowd. A High Performance Computing program is a low-level programming task that might consist of C/C++ or Fortran code, some shell scripts, OpenMP/MPI, and a high level of skill to put it all together.

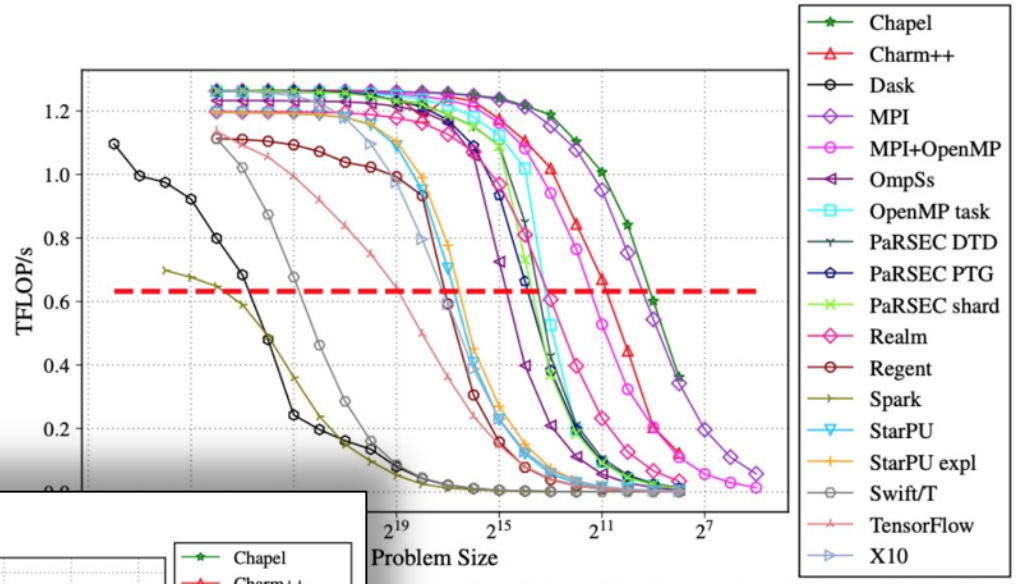
Chapel makes it easier by providing higher level language constructs for parallel computing that are similar to languages like Python or Matlab. All of the things that make HPC a hard nut to crack in C are handled at a higher level in Chapel—things like creating a distributed array spanning thousands of nodes, a namespace available on any node, and concurrency and parallel primitives.

HPC has always been somewhat of a niche. Partly because the need previously wasn't there, and partly because the skills were rare. Chapel brings the possibility of running machine learning algorithms at very large scale to the general software programmer. If nothing else, there's value for everyone in understanding the ideas and concepts that are surfaced elegantly in this language.

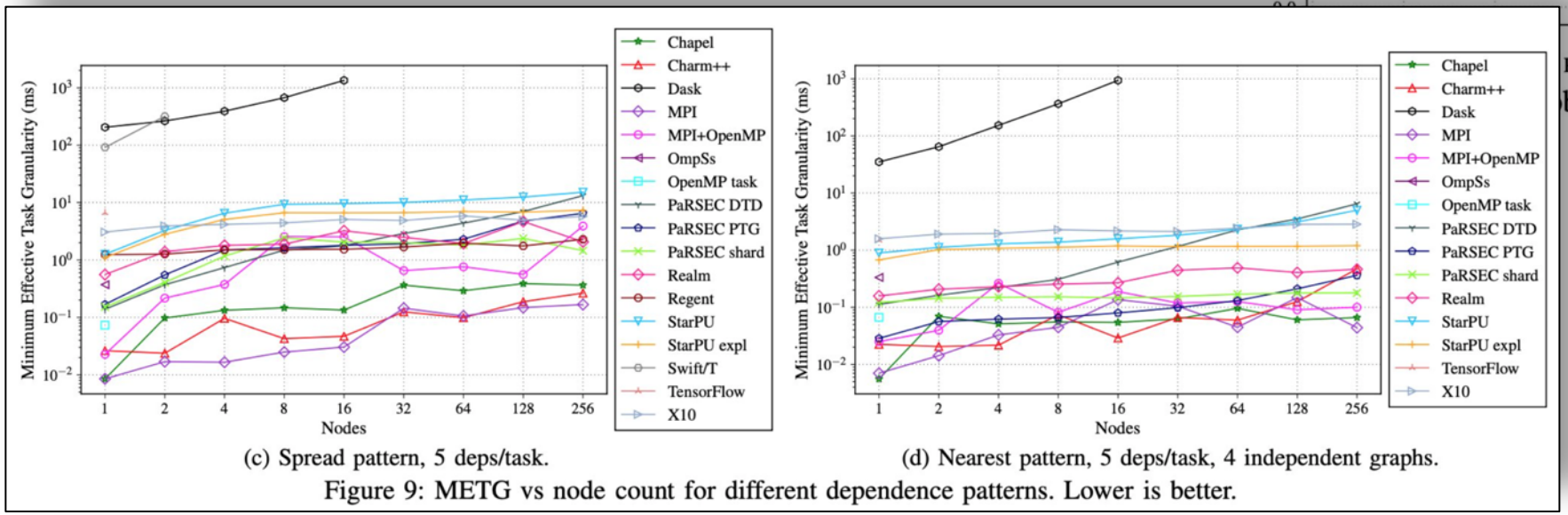
— *Steven Nuñez*

OUTREACH / COMMUNITY HIGHLIGHTS

- Nov: **Elliott Slaughter (Stanford) SC20 paper/talk**
  - *Task Bench: A Parameterized Benchmark for Evaluating Parallel Runtime Performance*
  - performed in-depth cross-runtime evaluation
  - Chapel compared quite well



blem size (stencil, 1 node). Higher is better.

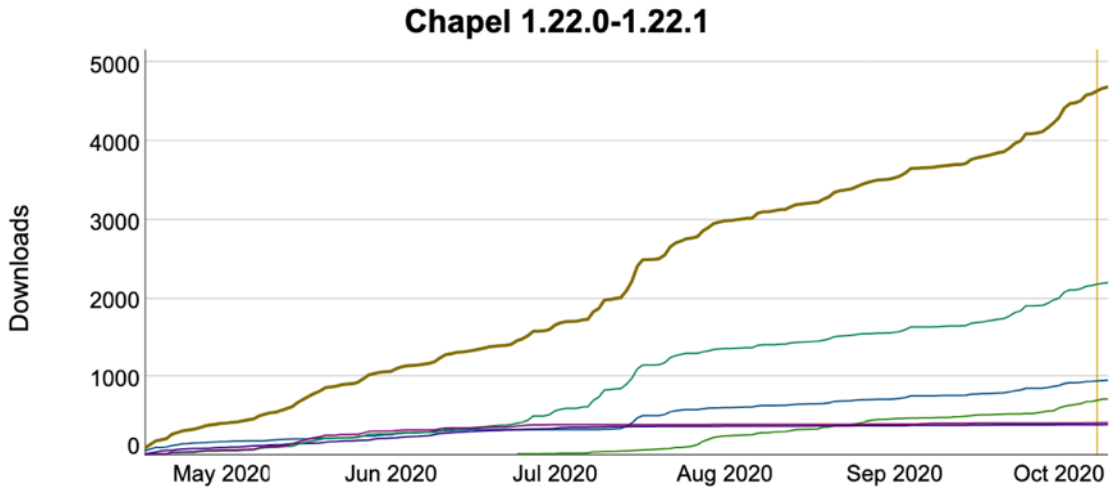


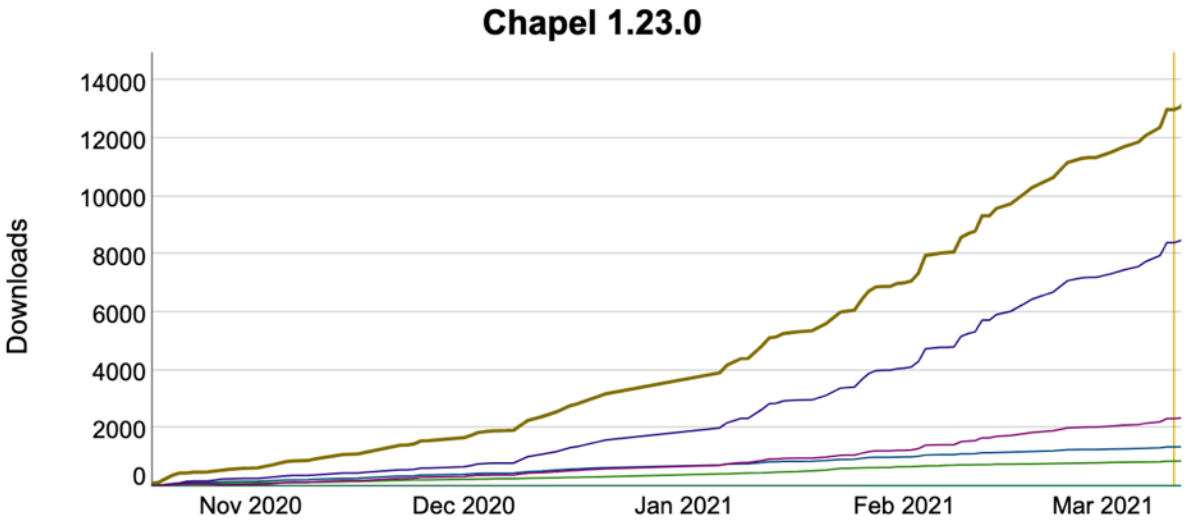
(c) Spread pattern, 5 deps/task. (d) Nearest pattern, 5 deps/task, 4 independent graphs.

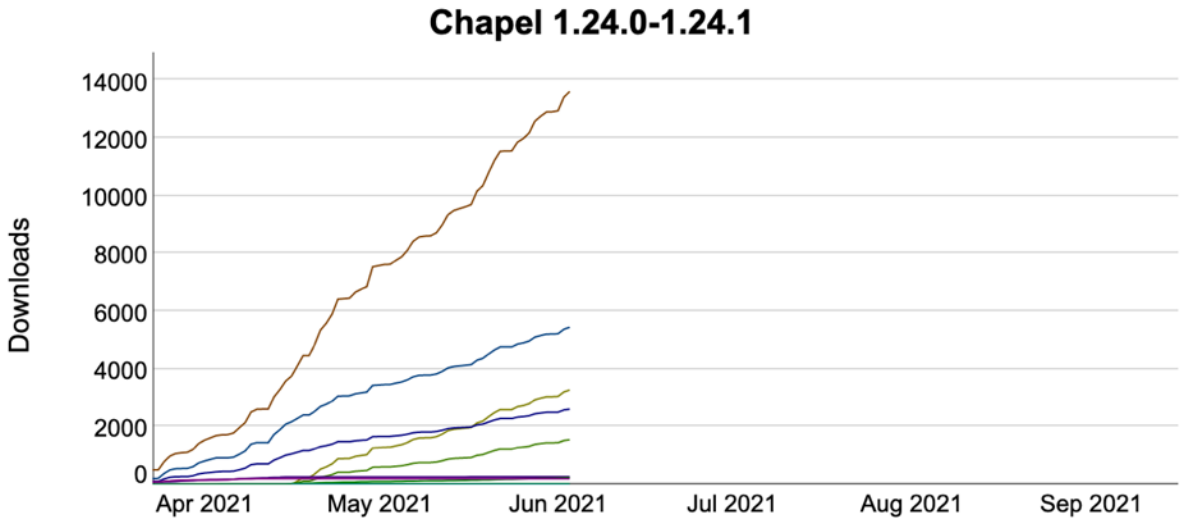
Figure 9: METG vs node count for different dependence patterns. Lower is better.

OUTREACH / COMMUNITY HIGHLIGHTS

- all-year: **spike in download counts**
  - from 4500–5500 for recent releases
  - to 13,000 for 1.23.0
  - to beating that in half the time for 1.24.0–1.24.1







IN MEMORIAM / COMMUNITY LOWLIGHTS

- Russel Winder, 12/30/55–1/23/21
  - A beloved figure and mentor in the UK programming community
  - A programming language enthusiast
  - A steady advocate for Chapel over many years
    - *"Fortran should be replaced with Chapel everywhere."*
    - *"And so we've got languages like Rust, D, and Chapel. And my contention is that, as Python programmers, we should not be rejecting these languages…we should actually be looking to use the right language for the right purpose at the right time."*
- Talks with Chapel ties:
  - *On Big Computation and Python*
    - PyCon UK 2017
  - *Fast Python? Don't Bother*
    - PyCon UK 2016
  - *Making Python Computations Fast*
    - PyCon UK 2015

WRAPPING UP

SUMMARY

Great progress since CHIUW 2020:

- Language stabilization is in good shape
- Significant performance and portability improvements
- Great user accomplishments and community interaction

Current Priorities:

- hiring to full headcount
- library stabilization
- compiler revamp
- GPU code-generation and vectorization
- more tuning for HPE Cray EX and InfiniBand
- launch Chapel blog / revamp website
- keep growing the user community

FOR YOUR CONSIDERATION

The 4th Annual Parallel Applications Workshop, Alternatives To MPI+X

Friday, November 19, 2021
8:30 am - 12:30 pm (CST)

Held in conjunction with SC21

SC21
St. Louis, Missouri | science beyond

Submission deadline July 23

(like CHIUW, accepts papers and talks)

https://sourceryinstitute.github.io/PAW/


THANK YOU
https://chapel-lang.org
@ChapelLanguage