

Hewlett Packard Enterprise

HPSF Project Proposal: The Chapel Programming Language

The Chapel Team HPSF Technical Advisory Council meeting January 9, 2025

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 and Productivity:

Chapel supports code that is as... ...**readable and writeable** as Python

- ...while also being as...
 - ...**fast** as Fortran / C / C++
 - ...scalable as MPI / SHMEM
 - ...**portable** as C
 - ...GPU-ready as CUDA / HIP / OpenMP / Kokkos / ...

Matches HPSF's focus on:

- lowering barriers to using HPC
- aiding HPC community growth
- enabling HPC development efforts
- portable software for diverse hardware
- performance and productivity



Performance and Productivity: Arkouda Argsort

HPE Cray EX

- Slingshot-11 network (200 Gb/s)
- 8192 compute nodes
- **256 TiB** of 8-byte values
- ~8500 GiB/s (~31 seconds)

HPE Cray EX

- Slingshot-11 network (200 Gb/s)
- 896 compute nodes
- 28 TiB of 8-byte values
- ~1200 GiB/s (~24 seconds)

HPE Apollo

- HDR-100 InfiniBand network (100 Gb/s)
- 576 compute nodes
- 72 TiB of 8-byte values
- ~480 GiB/s (~150 seconds)

Arkouda Argsort Performance



Implemented using ~100 lines of Chapel

Applications of Chapel



CHAMPS: 3D Unstructured CFD Laurendeau, Bourgault-Côté, Parenteau, Plante, et al. École Polytechnique Montréal





Arkouda: Interactive Data Science at Massive Scale Mike Merrill, Bill Reus, et al. U.S. DoD



Lattice-Symmetries: a Quantum Many-Body Toolbox Desk dot chpl: Utilities for Environmental Eng.

Tom Westerhout Radboud University



Chapel-based Hydrological Model Calibration Marjan Asgari et al. *University of Guelph*

Nelson Luis Dias The Federal University of Paraná, Brazil



Arachne Graph Analytics Bader, Du, Rodriguez, et al. New Jersey Institute of Technology



ChOp: Chapel-based Optimization T. Carneiro, G. Helbecque, N. Melab, et al. *INRIA, IMEC, et al.*



RapidQ: Mapping Coral Biodiversity Rebecca Green, Helen Fox, Scott Bachman, et al. The Coral Reef Alliance



CrayAl HyperParameter Optimization (HPO) Ben Albrecht et al. *Cray Inc. / HPE*



ChplUltra: Simulating Ultralight Dark Matter Nikhil Padmanabhan, J. Luna Zagorac, et al. *Yale University et al.*



ChapQG: Layered Quasigeostrophic CFD Ian Grooms and Scott Bachman University of Colorado, Boulder et al.



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

Productivity Across Diverse Application Scales (code and system size)



Computation: Aircraft simulation / CFD **Code size:** 100,000+ lines **Systems:** Desktops, HPC systems

	Par	600		
5	L		N	
	1		1	K

7 Questions for Éric Laurendeau: Computing Aircraft Aerodynamics in Chapel Posted on September 17, 2024.

Tags: Computational Fluid Dynamics User Experiences Interviews By: Engin Kayraklioglu, Brad Chamberlain

"Chapel worked as intended: the code maintenance is very much reduced, and its readability is astonishing. This enables undergraduate students to contribute, something almost impossible to think of when using very complex software."



Computation: Coral reef image analysis Code size: ~300 lines Systems: Desktops, HPC systems w/ GPUs



7 Questions for Scott Bachman: Analyzing Coral Reefs with Chapel Posted on October 1, 2024. Tags: Earth Sciences Image Analysis GPU Programming User Experiences Interviews

By: Brad Chamberlain, Engin Kayraklioglu

In this second installment of our <u>Seven</u> Questions for Chapel Users series, we're looking at a recent success story in which Scott Bachman used Chapel to unlock new scales of biodiversity analysis in coral reefs to study ocean health using satellite image processing. This is work that

"With the coral reef program, I was able to speed it up by a factor of 10,000. Some of that was algorithmic, but Chapel had the features that allowed me to do it."



Computation: Atmospheric data analysis Code size: 5000+ lines Systems: Desktops w/ GPUs



7 Questions for Nelson Luís Dias: Atmospheric Turbulence in Chapel Posted on October 15, 2024.

Tags: User Experiences Interviews Data Analysis

Computational Fluid Dynamics

By: Engin Kayraklioglu, Brad Chamberlain

In this edition of our <u>Seven Questions for Chapel Users</u> series, we turn to Dr. Nelson Luis Dias from Brazil who is using <u>Chapel to analyze data generated</u> by the <u>Amazon Tall Tower Observatory</u> (ATTO), a project dedicated to long-term, 24/7 monitoring of greenhouse gas fluctuations. Read

"Chapel allows me to use the available CPU and GPU power efficiently without low-level programming of data synchronization, managing threads, etc."

Where Does Chapel Run?

In the Browser:

- GitHub Codespaces
- Attempt This Online (ATO)

Laptops/Desktops:

- Linux/UNIX
- Mac OS X
- Windows (leveraging WSL)

HPC Systems:

- Commodity clusters
- HPE/Cray supercomputers, such as:
 - Frontier
 - Perlmutter
 - Piz Daint
 - Polaris
 - ...
- Other vendors' supercomputers

Cloud:

- AWS
- Microsoft Azure (?)
- Google Cloud (?)

CPUs:

- Intel
- AMD
- Arm (M1/M2, Graviton, A64FX, Raspberry Pi, ...)

GPUs:

- NVIDIA
- AMD

Networks:

- Slingshot
- Aries/Gemini
- InfiniBand
- AWS EFA
- Ethernet



Where does Chapel run?

https://chapel-lang.org/docs/usingchapel/portability.html

Synergies Between Chapel and HPSF's Goals

Lowering barriers to using HPC	 Chapel is helping users write real applications Many are writing HPC code for the first time others are simply leveraging their desktop multicore CPUs + GPUs 				
Aiding HPC community growth					
Enabling HPC development efforts	 Others are HPC experts, working more quickly than they otherwise could've 				
Portable software for diverse hardware	 Chapel currently supports most any HPC, desktop, or cloud system Its language design and code architecture support porting to others 				
Performance and productivity	 Chapel performance often matches or beats conventional HPC technologies Code is almost always shorter and easier to read/write/maintain 				

How Is Chapel Developed?

Platform: GitHub License: Apache 2.0 Release Cadence: Quarterly (Mar, June, Sept, Dec) Contributors:

- over time: 200+ from 100+ affiliations worldwide
- per-release: ~25–35, primarily from HPE/Cray
- docs for contributors: chapel-lang.org/docs/developer/

Code of Conduct: CODE_OF_CONDUCT.md

Governance: HPE-led, with user input and guidance **Decision-Making:**

- consensus-oriented
- ad hoc subteams to explore and propose solutions
- discussions on GitHub issues and in community forums

Product × Solutions × Resou	urces 🗸 Open Source 🗸 Enterprise 🗸 Pricing	Q Sign in Sign up
	Ų	Notifications $\begin{array}{c} \begin{array}{c} \begin{array}{c} \begin{array}{c} \\ \\ \\ \end{array} \end{array}$ Fork 424 $\begin{array}{c} \end{array}$ Star 1.8k
<> Code ③ Issues 2.7k \$\$ Pull re	equests 102 🕟 Actions 🖽 Projects 😲 Security 🗠 Insights	
د الله الله المعادم المعا معادم المعادم المعادم المعادم المعادم المعادم المعادم المعادم معادم معادم معادم معادم معادم معادم معادم المعادم م	Q Go to file <> Code	- About
riftEmber Fix mistakes from removing	FILENAME_MAX (#26491) 🚥 🔸 5e5a0e9 · now 🕚 105,362 Commit	a Productive Parallel Programming Language
igithub	Handle PRs not based off recent main in large files c 3 weeks as	go c ² chapel-lang.org
compiler	Replace variable-length char array with std::string 1 hour ag	go language programming-language
doc	Auto-generate documentation for chplcheck lint rul 28 minutes as	open-source performance compiler
frontend	Add tests for type method iterators 18 hours as	parallel-computing distributed-computing
highlight	add missing keywords to chpl-mode.el 3 months ag	go high-performance-computing chapel
🖿 make	Update copyrights to 2025 with fixed script 2 days ag	go productive
📄 man	Update copyrights to 2025 with fixed script 2 days as	go 🖾 Readme
modules	use this.check() 1 hour ag	회 전체 Unknown and 2 other licenses found 영 ⓒ Code of conduct
📄 runtime	Fix discarding cstring literal constness 1 hour ag	go -Vr Activity
🖿 test	Auto-generate documentation for chplcheck lint rul 28 minutes as	□ Custom properties
🖿 third-party	unrevert io changes 2 days ag	go 📀 64 watching
tools	Auto-generate documentation for chplcheck lint rul 28 minutes as	go % 424 forks
🖿 util	Auto-generate documentation for chplcheck lint rul 28 minutes as	30
🗋 .dockerignore	Convert Dockerfile into multiple stages, delete large 11 months ag	go Releases 34
🗋 .gitignore	Use a subdirectory .gitignore for rust ignores last ye	ar Chapel 2.3.0 Release (Winter Latest)

https://github.com/chapel-lang/chapel

How Is Chapel Deployed?

Release Formats:

- Source releases via GitHub
- Spack
- E4S
- Linux packages via apt/rpm
- Homebrew
- Docker
- Modules on HPE Cray systems
- ATO / GitHub Codespaces

Chapel releases leverage and bundle:

- GASNet (LBNL) and libfabric (OFI) for communication
- **Qthreads** (Sandia) for tasking
- hwloc (OpenMPI) for HW introspection
- **jemalloc** for memory allocation
- **LLVM** for back-end compilation
- **GMP** for bigint support
- **re2** for regular expression support
- libunwind, utf8-decoder, whereami

	EL.		DOWNLOAD	DOCS -	LEARN	RESOURCES -	COMMUNIT
DOWN		CHAPEL					
Downloa	ading from Sour	ce▼					
To downlo	oad and install Chapel from s	source, download chapel-2.3.0.tar.gz from (GitHub, then unpack an	nd build it a	s described in	the Quickstart inst	tructions.
Downlo	ading with Spac	k-					
201110	ading man opaci						
To get sta	irted with the Chapel Spack p	backage:					
1. Inst 2. To c Cha	tall the Spack package mana customize the Chapel install apel's documentation can be	ager on your system, if it isn't already there. ation, use the variants of the Chapel Spack set using variants of the Chapel Spack pack	package as opposed to kage.	o the norma	al CHPL_* en	vironment variables	s. Most settings de
3. Inst	tall the Chapel package, spe	citying any variant desired. For example, to a	also install the chpidoc	tool, use	spack insta	ll chapel+chpldo	DC.
Downloa Downloa	ading with Dock	er > ebrew >					
Downlos Downlos Downlos Installin We provid 1. Dov	ading with Dock ading with Home ading on HPE Sy ng with Linux Pac le Chapel packages for sever wnload the package for your	er > ebrew > rstems > ckage Managers ~ ral different Linux distributions, though they system using one of the following links:	come with some perfo	prmance ca	veats. They ca	an be installed as fo	ollows:
Downloa Downloa Downloa Installin We provid 1. Dov	ading with Dock ading with Home ading on HPE Sy ng with Linux Pac le Chapel packages for sever wnload the package for your Operating System	er > ebrew > vstems > ckage Managers vstem using one of the following links: Single-Locale Configuration	come with some perfo	ormance ca	veats. They ca	an be installed as fo	ollows:
Downloa Downloa Downloa Installin We provid 1. Dow	ading with Dock ading with Home ading on HPE Sy og with Linux Pac le Chapel packages for sever writoad the package for your Operating System maLinux 9	er > ebrew > estems > ckage Managers ral different Linux distributions, though they system using one of the following links: Single-Locale Configuration [x86_64] [arm64]	GASNet+L [x86_64] [arm6+	prmance ca JDP 4]	veats. They cr Sit [x86_64] [t	an be installed as f arm+OFI arm64]	ollows:
Downloa Downloa Downloa Installin We provid 1. Dov	ading with Dock ading with Home ading on HPE Sy ng with Linux Pac le Chapel packages for sever wiload the package for your Operating System maLinux 9 nazon Linux 2023	er > ebrew > vstems > ckage Managers ral different Linux distributions, though they system using one of the following links: Single-Locale Configuration [x86_64] [arm64] [x86_64] [arm64]	Come with some perfor GASNet+L [x86_64] [arm6- [x86_64] [arm6-	Drmance ca JDP 4] 4]	veats. They cr [x86_64] [t [x86_64] [t	an be installed as f urm+OFI arm64] arm64]	ollows:
Downloa Downloa Downloa Installin We provid 1. Dev	ading with Dock ading with Home ading on HPE Sy ng with Linux Pac le Chapel packages for sever wiload the package for your Operating System maLinux 9 nazon Linux 2023 abian 11	er > ebrew > vstems > ckage Managers ral different Linux distributions, though they system using one of the following links: Single-Locale Configuration [x86_64] [arm64] [x86_64] [arm64] [x86_64] [arm64] [x86_64] [arm64]	Come with some perfor GASNet+L [x86_64] [arm6- [x86_64] [arm6- [x86_64] [arm6-	Drmance ca JDP 4] 4] 4]	veats. They cr Sk [x86_64] [r [x86_64] [r	an be installed as fo rrm+0FI arm64]	ollows:
Downloa Downloa Downloa Installin We provid 1. Dov	ading with Dock ading with Home ading on HPE Sy og with Linux Pac le Chapel packages for sever wiload the package for your Operating System maLinux 9 nazon Linux 2023 abian 11 abian 12	er > ebrew > vstems > ckage Managers > c	Come with some perfor GASNet+L [x86_64] [arm6- [x86_64] [arm6- [x86_64] [arm6- [x86_64] [arm6-	prmance ca JDP 4] 4] 4] 4]	veats. They cr Sk [x86_64] [z [x86_64] [z	an be installed as fo Irm+OFI Irm64] Irm64]	ollows:
Downloa Downloa Downloa Installin We provid 1. Dov	ading with Dock ading with Home ading on HPE Sy ong with Linux Pac le Chapel packages for sever write adding on HPE Sy ong with Linux Pac le Chapel package for your operating System maLinux 9 nazon Linux 2023 adding 11 adding 12 dora 40	er > ebrew > vstems > ckage Managers > c	GASNet+L [x86_64] [arm6- [x86_64] [arm6- [x86_64] [arm6- [x86_64] [arm6- [x86_64] [arm6-	DIPP 4] 4] 4] 4] 4] 4]	veats. They c: Sit [x86_64] [t [x86_64] [t	an be installed as fo urm+OFI arm64] arm64]	ollows:
Downloa Downloa Downloa Installin We provid 1. Dov	ading with Dock ading with Home ading on HPE Sy ong with Linux Pac le Chapel packages for sever write adding on HPE Sy ong with Linux Pac le Chapel package for your operating System maLinux 9 nazon Linux 2023 adding 11 adding 12 dora 40 dora 41	er > ebrew > vstems > vstems > vstems > vstems = vstems = vstems = vstems = vstems = vstems = vstem using one of the following links: vstem using one of the following	Come with some perfor GASNet+L [x86_64] [arm64 [x86_64] [arm64 [x86_64] [arm64 [x86_64] [arm64 [x86_64] [arm64 [x86_64] [arm64	DPP 4] 4] 4] 4] 4] 4] 4]	veats. They cr Sit [x86_64] [t [x86_64] [t	an be installed as fo urm+OFI arm64] arm64]	ollows:
Downloa Downloa Downloa Installin We provid 1. Dov	ading with Dock ading with Home ading on HPE Sy ading on HPE Sy by the second second ading on HPE Sy adjust the second second second second second ading on HPE Sy adjust the second second second second second adjust the second	er > ebrew > vstems > vstem using one of the following links: vsystem using one of the following links: vsystem using one of the following links: vstem using one of the following link	Come with some perfor GASNet+L [x86_64] [arm64 [x86_64] [arm64 [x86_64] [arm64 [x86_64] [arm64 [x86_64] [arm64 [x86_64] [arm64 [x86_64] [arm64	DPP 4] 4] 4] 4] 4] 4] 4] 4] 4]	veats. They cr Sit [x86_64] [z [x86_64] [z [x86_64] [z	an be installed as fo arm64] arm64] arm64]	ollows:
Downloa Downloa Downloa Installin We provid 1. Dow Arr Arr Arr Arr Arr Be Be Fee Fee RH Ro	ading with Dock ading with Home ading on HPE Sy ading on HPE Sy ading on HPE Sy ading on HPE Sy ading with Linux Pac be chapel packages for sever who ad the package for your operating System maLinux 9 mazon Linux 2023 abian 11 abian 12 dora 40 dora 41 HEL 9 motyLinux 9	er > ebrew > vstems > vstems > ckage Managers > ckage & ckage Managers > ckage & ckage	Come with some perfor GASNet+L [x86_64] [arm6- [x86_64] [arm6- [x86_64] [arm6- [x86_64] [arm6- [x86_64] [arm6- [x86_64] [arm6- [x86_64] [arm6- [x86_64] [arm6- [x86_64] [arm6-	DPP 4] 4] 4] 4] 4] 4] 4] 4] 4] 4] 4]	veats. They co [x86_64] [c [x86_64] [c [x86_64] [c [x86_64] [c	an be installed as for arm64] arm64] arm64]	ollows:
Downloa Downloa Downloa Installin We provid 1. Dow Alr An De De Fee Fee RH Ro Ub	ading with Dock ading with Home ading on HPE Sy ading on HPE S	er > ebrew > estems >	Come with some performance of the some some some some some some some som	DP 4] 4] 4] 4] 4] 4] 4] 4] 4] 4]	veats. They cr [x86_64] [r [x86_64] [r [x86_64] [r [x86_64] [r [x86_64] [r [x86_64] [r	an be installed as for arm64] arm64] arm64] arm64] arm64]	ollows:

https://chapel-lang.org/download/

How Is Chapel Tested?

CI/CD: relatively quick, pre-merge checks **Smoke Testing:** longer, post-merge checks

goal: head off catastrophes overnight

Nightly Testing: 125+ jobs managed with Jenkins

- leverages a suite of 17,500+ tests
- spans a multitude of platforms, vendors, networks, ...

None

Filter

- jobs for specific configs, compiler flags, user apps, ...
- jobs for memory leaks, valgrind/asan errors, ...
- jobs for correctness and performance tracking

🏟 Jenkins			Q sear	ch		?	g in
Jenkins > correctness-test-memleaks	#1716 First Results						
Back to Project Status Changes Console Output View as plain text View Build Information	Test Result 5 failures (+5), 1,717 skipped (+2) All Failed Tests					6,982 tests <u>Took 1 day</u>	: (+11) <u>/ 0 hr.</u>
listory	Test Name					Duration .	Age
Timings	classes/initializers/postInit.throwing-superclass	🗦 (Failed 1 times in the last 1 run	s. Flakiness: 0	%, Stability: 0%)	5.4 sec	1
Environment Variables	classes/initializers/postInit.throwing-superclass2	🔅 (Failed 1 times in the last 1 ru	ns. Flakiness: (%, Stability: 0	%)	5.5 sec	1
Git Build Data	classes/initializers/postInit.throwing_superclass4	🔅 (Failed 1 times in the last 1 ru	ıns. Flakiness: (%, Stability: 09	%)	5.5 sec	1
Test Result	classes/initializers/postInit.throwing	imes in the last 30 runs. Flakines	ss: 3%, Stability	r: 96%)		5.5 sec	1
Support	runtime/jhh.numColocales2	he last 1 runs. Flakiness: 0%, Sta	ability: 0%)			5.9 sec	1
🖕 Previous Build	All Tests						
🔶 Next Build	Package	Duration	Fail (diff)	Skip (diff)	Pass (diff)	Total	(diff)
	(root)	1 day 0 hr	5 +5	1717 +2	15256 +4	16978	+11
	sparse/s	10 sec	0	0	2	2	
	<u>users/christopher</u>	11 sec	0	0	2	2	





10

How Does the Chapel Community Communicate?

Live/Virtual Events

- <u>ChapelCon</u> (formerly CHIUW), annually
- Office Hours, monthly
- Live Demo Sessions, monthly

<u>Community / User Forums</u>

- Discord
- **Discourse**
- Email Contact Alias
- GitHub Issues
- Gitter
- <u>Reddit</u>
- Stack Overflow

Discord

()

- 😳 reddit

chapel+qs@discoursemail.com

III GITTER

- stack overflow

Electronic Broadcasts

- <u>Chapel Blog</u>, ~biweekly
- <u>Community Newsletter</u>, quarterly
- <u>Announcement Emails</u>, around big events

Social Media

- Bluesky
- Facebook
- Linked in • LinkedIn
- mastodon <u>Mastodon</u>
- <u>X / Twitter</u> X
- <u>YouTube</u> **Particular**

Chapel's HPSF Application

Maturity Level:

- We're applying at the "Established" stage, believing we easily meet the requirements
- We're interested in improving our processes under HPSF, which would also help move us toward the "Core" stage
 - establishing a community governing body and documenting it
 - establishing security processes
 - extending merge privileges to non-HPE developers

Motivations for Applying:

- Believe it will help improve our project's visibility and stature
- Expect it to help address "single-vendor" concerns that discourage potential users and collaborators
- Hope to network with other open-source HPC projects and share best practices
 - Our experience with HPC testing, portability, performance tracking, etc. may be useful to other projects, and we have lots to learn too
 - We're particularly interested in leveraging Linux Foundation / HPSF expertise in the "Core" areas noted above

Infrastructure Needs:

- Nothing pressing at present
- Ideally: Compute resources outside HPE for community testing or the ability to "try Chapel in the cloud"
- Might be nice: Assistance with things like paid CI/CD runners, DNS registration, financial donations, etc.

Thank you

https://chapel-lang.org @ChapelLanguage

