Reflections on 30 Years of HPC Programming Models (Abridged)

Brad Chamberlain

CLSAC 2025

October 8, 2025

30 Years Ago vs. Now: Top HPC Systems

Top 5 systems in the Top500, June 1995:

- **Cores:** 80–3680 cores
- **Rmax:** ~98.9–170 GFlop/s
- **Systems:** Fujitsu, Intel Paragon XP/S, Cray T3D
- **Networks:** crossbar, mesh, 3D torus

TOP500 LIST - JUNE 1995

R_{max} and R_{peak} values are in GFlop/s. For more details about other fields, check the TOP500 description.

R_{peak} values are calculated using the advertised clock rate of the CPU. For the efficiency of the systems you should take into account the Turbo CPU clock rate where it applies.

← 1-100 101-200 201-300 301-400 401-500 →

Rank	System	Cores	Rmax (GFlop/s)	Rpeak (GFlop/s)	Power (kW)
1	Numerical Wind Tunnel, Fujitsu National Aerospace Laboratory of Japan Japan	140	170.00	235.79	
2	XP/S140, Intel Sandia National Laboratories United States	3,680	143.40	184.00	
3	XP/S-MP 150, Intel DOE/SC/Oak Ridge National Laboratory United States	3,072	127.10	154.00	
4	T3D MC1024-8, Cray/HPE Government United States	1,024	100.50	153.60	
5	VPP500/80, Fujitsu National Lab. for High Energy Physics Japan	80	98.90	128.00	

Top 5 systems in the Top 500, June 2025:

- **Cores:** 2,073,600–11,039,616 **(~563x–138,000x)**
- **Rmax:** ~477.9–1742.0 PFlop/s **(~2,810,000x–17,600,000x)**
- **Systems:** HPE Cray EX, Eviden Bullsequana, Microsoft Azure
- **Networks:** Slingshot-11, InfiniBand NDR

TOP500 LIST - JUNE 2025

R_{max} and R_{peak} values are in PFlop/s. For more details about other fields, check the TOP500 description.

R_{peak} values are calculated using the advertised clock rate of the CPU. For the efficiency of the systems you should take into account the Turbo CPU clock rate where it applies.

← 1-100 101-200 201-300 301-400 401-500 →

Rank	System	Cores	Rmax (PFlop/s)	Rpeak (PFlop/s)	Power (kW)
1	**El Capitan** - HPE Cray EX255a, AMD 4th Gen EPYC 24C 1.8GHz, AMD Instinct MI300A, Slingshot-11, TOSS, HPE DOE/NNSA/LLNL United States	11,039,616	1,742.00	2,746.38	29,581
2	**Frontier** - HPE Cray EX235a, AMD Optimized 3rd Generation EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11, HPE Cray OS, HPE DOE/SC/Oak Ridge National Laboratory United States	9,066,176	1,353.00	2,055.72	24,607
3	**Aurora** - HPE Cray EX - Intel Exascale Compute Blade, Xeon CPU Max 9470 52C 2.4GHz, Intel Data Center GPU Max, Slingshot-11, Intel DOE/SC/Argonne National Laboratory United States	9,264,128	1,012.00	1,980.01	38,698
4	**JUPITER Booster** - BullSequana XH3000, GH Superchip 72C 3GHz, NVIDIA GH200 Superchip, Quad-Rail NVIDIA InfiniBand NDR200, RedHat Enterprise Linux, EVIDEN EuroHPC/FZJ Germany	4,801,344	793.40	930.00	13,088
5	**Eagle** - Microsoft NDv5, Xeon Platinum 8480C 48C 2GHz, NVIDIA H100, NVIDIA Infiniband NDR, Microsoft Azure Microsoft Azure United States	2,073,600	561.20	846.84	

30 Years Ago vs. Now: Top HPC Systems

Top 5 systems in the Top500, June 1995:

- **Cores:** 80–3680 cores
- **Rmax:** ~98.9–170 GFlop/s
- **Systems:** Fujitsu, Intel Paragon XP/S, Cray T3D
- **Networks:** crossbar, mesh, 3D torus

HPC HW has become far more capable…

Top 5 systems in the Top 500, June 2025:

- **Cores:** 2,073,600–11,039,616 **(~563x–138,000x)**
- **Rmax:** ~477.9–1742.0 PFlop/s **(~2,810,000x–17,600,000x)**
- **Systems:** HPE Cray EX, Eviden Bullsequana, Microsoft Azure
- **Networks:** Slingshot-11, InfiniBand NDR

TOP500 LIST - JUNE 1995

R_{max} and R_{peak} values are in GFlop/s. For more details about other fields, check the TOP500 description.

R_{peak} values are calculated using the advertised clock rate of the CPU. For the efficiency of the systems you should take into account the Turbo CPU clock rate where it applies.

← 1-100 101-200 201-300 301-400 401-500 →

Rank	System			
1	Numerical Wind Tunnel, Fujitsu National Aerospace Laboratory of Japan Japan			
2	XP/S140, Intel Sandia National Laboratories United States			
3	XP/S-MP 150, Intel DOE/SC/Oak Ridge National Laboratory United States			
4	T3D MC1024-8, Cray/HPE Government United States			
5	VPP500/80, Fujitsu National Lab. for High Energy Physics Japan	80	98.90	128.00

And complex!

- **commodity vector processors**
- **multicore processors**
- **multi-socket compute nodes**
- **NUMA compute node architectures**
- **high-radix, low-diameter interconnects**
- **GPU computing**

(Often in ways that hurt programmability)

TOP500 LIST - JUNE 2025

R_{max} and R_{peak} values are in PFlop/s. For more details about other fields, check the TOP500 description.

R_{peak} values are calculated using the advertised clock rate of the CPU. For the efficiency of the systems you should take into account the Turbo CPU clock rate where it applies.

← 1-100 101-200 201-300 301-400 401-500 →

System	Cores	Rmax (PFlop/s)	Rpeak (PFlop/s)	Power (kW)
…tan - HPE Cray EX255a, AMD 4th Gen EPYC 24C …D Instinct MI300A, Slingshot-11, TOSS, HPE …LNL	11,039,616	1,742.00	2,746.38	29,581
…235a, AMD Optimized 3rd …z, AMD Instinct MI250X, …PE …oratory	9,066,176	1,353.00	2,055.72	24,607
…xascale Compute Blade, …GHz, Intel Data Center GPU …ional Laboratory	9,264,128	1,012.00	1,980.01	38,698
…ooster - BullSequana XH3000, GH Superchip …z, NVIDIA GH200 Superchip, Quad-Rail NVIDIA …Band NDR200, RedHat Enterprise Linux, EVIDEN …roHPC/FZJ Germany	4,801,344	793.40	930.00	13,088
Eagle - Microsoft NDv5, Xeon Platinum 8480C 48C 2GHz, NVIDIA H100, NVIDIA Infiniband NDR, Microsoft Azure Microsoft Azure United States	2,073,600	561.20	846.84	

30 Years Ago vs. Now: Top HPC Systems and Programming Notations

Top 5 systems in the Top500, June 1995:

- **Cores:** 80–3680 cores
- **Rmax:** ~98.9–170 GFlop/s
- **Systems:** Fujitsu, Intel Paragon XP/S, Cray T3D
- **Networks:** crossbar, mesh, 3D torus

HPC HW has become far more capable…

Top 5 systems in the Top 500, June 2025:

- **Cores:** 2,073,600–11,039,616 **(~563x–138,000x)**
- **Rmax:** ~477.9–1742.0 PFlop/s **(~2,810,000x–17,600,000x)**
- **Systems:** HPE Cray EX, Eviden Bullsequana, Microsoft Azure
- **Networks:** Slingshot-11, InfiniBand NDR

Broadly-adopted HPC programming notations:

- **Languages:** C, C++, Fortran
- **Inter-node:** MPI, SHMEM
- **Intra-node:** vendor-specific pragmas & intrinsics
  - OpenMP on the horizon: 1997
- **Scripting:** Perl, [[t]c]sh, Tcl/TK

Broadly-adopted HPC programming notations:

- **Languages:** C, C++, Fortran
- **Inter-node:** MPI, SHMEM, Fortran 2008 Coarrays
- **Intra-node:** OpenMP, vendor-specific pragmas & intrinsics
- **GPUs:** CUDA, HIP, SYCL, Kokkos, OpenMP, OpenACC, …
- **Scripting:** Python, bash

30 Years Ago vs. Now: Top HPC Systems and Programming Notations

Top 5 systems in the Top500, June 1995:

- **Cores:** 80–3680 cores
- **Rmax:** ~98.9–170 GFlop/s
- **Systems:** Fujitsu, Intel Paragon XP/S, Cray T3D
- **Networks:** crossbar, mesh, 3D torus

HPC HW has become far more capable…

Top 5 systems in the Top 500, June 2025:

- **Cores:** 2,073,600–11,039,616 **(~563x–138,000x)**
- **Rmax:** ~477.9–1742.0 PFlop/s **(~2,810,000x–17,600,000x)**
- **Systems:** HPE Cray EX, Eviden Bullsequana, Microsoft Azure
- **Networks:** Slingshot-11, InfiniBand NDR

Broadly-adopted HPC programming notations:

- **Languages:** C, C++, Fortran
- **Inter-node:** MPI, SHMEM
- **Intra-node:** vendor-specific pragmas & intrinsics
  - OpenMP on the horizon: 1997
- **Scripting:** Perl, [[t]c]sh, Tcl/TK

…while HPC notations have largely stayed the same, modulo GPU computing

Broadly-adopted HPC programming notations:

- **Languages:** C, C++, Fortran
- **Inter-node:** MPI, SHMEM, Fortran 2008 Coarrays
- **Intra-node:** OpenMP, vendor-specific pragmas & intrinsics
- **GPUs:** CUDA, HIP, SYCL, Kokkos, OpenMP, OpenACC, …
- **Scripting:** Python, bash

Why the relative stasis in HPC programming languages?

Is it because language design is dead?

"Programming language design ceased to be relevant in the 1980s."

—anonymous reviewer, circa 1995 (paraphrased, from memory)

Seems unlikely...

- Consider all the currently relevant languages that emerged or rose to prominence during those 30 years:
  - **Java** (~1995)
  - **Javascript** (~1995)
  - **Python** (~1989; v2.0 ~2000)
  - **C#** (~2000)
  - **Go** (~2009)
  - **Rust** (~2012)
  - **Julia** (~2012)
  - **Swift** (~2014)

Such languages have become favorite day-to-day languages of many users across multiple disciplines

Why the relative stasis in HPC programming languages?

Is it for lack of trying?

Again "no"…

- **Mid-to-late 90's Classics:**
  - HPF: High Performance Fortran
  - ZPL
  - NESL
- **PGAS founding members:**
  - CAF: Coarray Fortran
  - UPC
  - Titanium
- **C-based approaches:**
  - Cilk
  - SAC: Single-Assignment C
- **HPCS-era languages:**
  - Chapel
  - Fortress
  - X10
  - CAF 2.0
- **Post-HPCS:**
  - XcalableMP
  - Regent
- **Embedded pseudo-languages**
  - Charm++, Global Arrays, HPX, UPC++, Legion, …
- **And many more…**

Not all attempts have been worthy of broad adoption;
yet, past failures to achieve broad adoption don't mean we should stop trying

Why the relative stasis in HPC programming languages?

Is it due to lack of added value?

Many would argue "no"…

- *"Why Languages Matter More Than Ever"* by Kathy Yelick, CHIUW 2018 keynote

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

Programming languages offer unique advantages over libraries and extensions

Why the relative stasis in HPC programming languages?

Q: So then why?

We are a unique community with unique computational needs

We often must focus on maintaining longstanding apps rather than writing new ones

We tend to invent new programming notations for each new form of HW parallelism

- commodity vectorization → vendor-specific pragmas and intrinsics
- distributed memory → MPI, SHMEM, Fortran 2008 Coarrays, UPC
- multicore → OpenMP
- GPUS → CUDA, HIP, SYCL, Kokkos, OpenMP, OpenACC, OpenCL, …

Our HW::SW investment and focus tilt heavily toward HW

We generally doubt that we are large / important enough to warrant and sustain a language of our own

We tend not to develop support structures for HPC software beyond the research stage

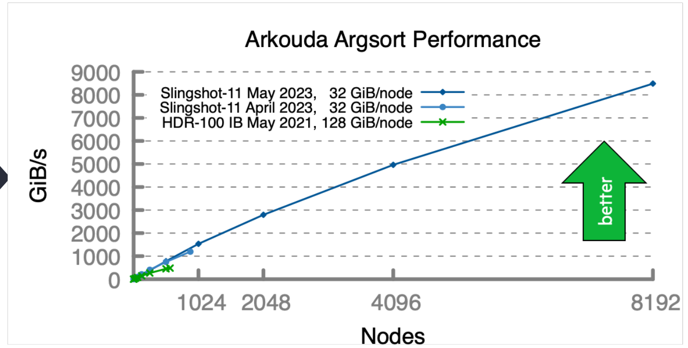
- Though maybe that's changing with HPSF…?

HPSF
HIGH PERFORMANCE
SOFTWARE FOUNDATION

Chapel's adaptable persistence

Chapel pre-dates all the architectural changes mentioned previously, other than commodity vectors…

- **commodity vector processors**
- **multicore processors**
- **multi-socket compute nodes**
- **NUMA compute node architectures**
- **high-radix, low-diameter interconnects**
- **GPU computing**



…yet it supports all of these HW features

- Moreover, using essentially the same language features as ~20 years ago
- How? By focusing on expressing parallelism and locality independently from HW mechanisms

Chapel's Generality

Chapel has proven to be generally applicable, as designed

- Read about user experiences in our *7 Questions with Chapel Users* interview series

Chapel Language Blog

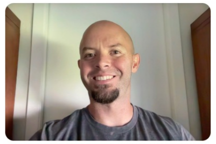
About Chapel Website Featured Series Tags Authors All Posts

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



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

7 Questions for Nelson Luís Dias: Atmospheric Turbulence in Chapel

Posted on October 15, 2024.

Tags: User Experiences | Interviews | Data Analysis | Earth Sciences | Computational Fluid Dynamics

By: Engin Kayraklioglu, Brad Chamberlain

7 Questions for David Bader: Graph Analytics at Scale with Arkouda and Chapel

Posted on November 6, 2024.

Tags: User Experiences | Interviews | Graph Analytics | Arkouda

By: Engin Kayraklioglu, Brad Chamberlain

7 Questions for Bill Reus: Interactive Supercomputing with Chapel for Cybersecurity

Posted on February 12, 2025.

Tags: User Experiences | Interviews | Data Analysis | Arkouda

By: Engin Kayraklioglu, Brad Chamberlain

7 Questions for Tiago Carneiro and Guillaume Helbecque: Combinatorial Optimization in Chapel

Posted on July 30, 2025.

Tags: User Experiences | Interviews

By: Engin Kayraklioglu, Brad Chamberlain

7 Questions for Marjan Asgari: Optimizing Hydrological Models with Chapel

Posted on September 15, 2025.

Tags: User Experiences | Interviews | Earth Sciences

By: Engin Kayraklioglu, Brad Chamberlain

AI, HPC, and Languages

Q: AI can program now*. Would we still benefit from better HPC languages?

My answer is "yes"...

- To say we no longer need good programming languages and compilers in the age of AI is like saying we no longer need to invest in roads, automobile manufacturing, fuel efficiency, safety, and traditional driving skills in an age of self-driving cars.

(* = your mileage may vary)

Closing Statements

*I consider HPC programmers—whether current or aspiring—
to be at least as worthy of modern languages as the
Python, Rust, Swift, and Julia communities*

*Within the next 30 years,
the number of broadly adopted scalable parallel languages
should be ≥1, rather than the current 0.*

The Advanced Programming Team at HPE

- Improving system design and operation via **system simulation** and **telemetry** at scale
- Making HPC speeds and scales accessible to all programmers with **Chapel**
- Making HPC speeds and scales available to Python programmers, interactively, with **Arkouda**
  - Growing Arkouda's strengths with **Honeycomb**: user-extensible, multi-lingual (Julia, Rust, plain English, ...)

Thank You

@ChapelLanguage