CAASCADE: Understanding HPC Applications for Evidence-based Co-design

Jack C. Wells, Director of Science, NCCS

M. Graham Lopez, Computer Scientist, Future Technologies (CSMD)

Oscar Hernandez, Computer Scientist, Computer Science Research (CSMD)

Reuben Budiardja, Computational Scientist, Scientific Computing (NCCS)

Jisheng Zhao, Computer Scientist, Georgia Tech University

Vivek Sarkar, Professor, Georgia Tech University

What's the problem?

Understanding the DOE HPC application landscape is important for effective integration of heterogeneous computational architectures and exploration of future computing technologies.

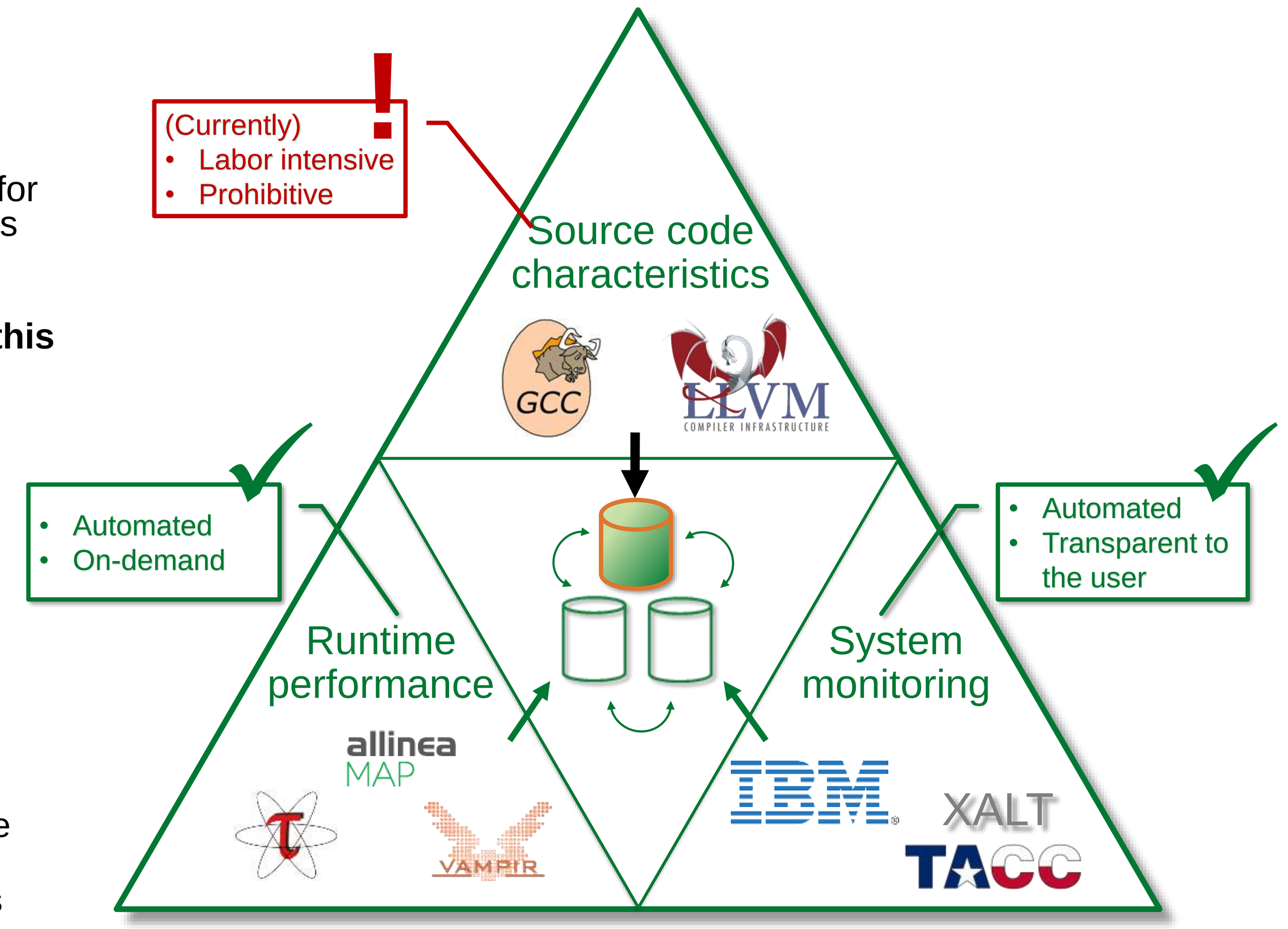
There is a paucity of quantitative data on critical application characteristics, and absence of tools to obtain and manage this data.

Examples:

- Which programming models are being used?
- Which programming model features are critical?
- How important is Fortran to the DOE?

Unanswered questions from CORAL Co-design:

- How should OpenACC support deep copy?
  - ✓ provide data structure shapes and layouts
- Which OpenMP features should be prioritized?
  - ✓ identify OpenMP constructs in use
- What kind of internode communications need enhanced hardware support?
  - ✓ identify prevalent MPI library calls and payload characteristics



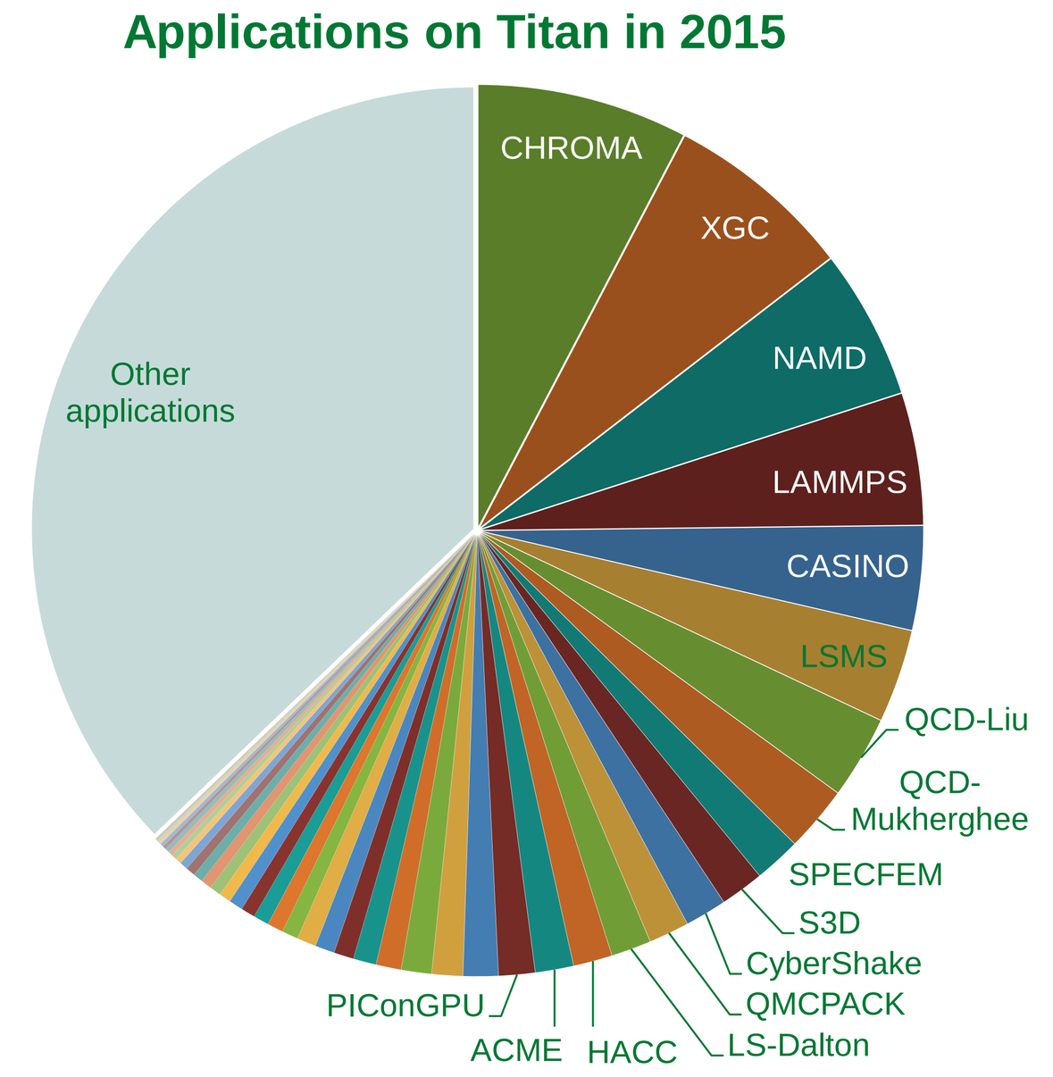
Currently available information about HPC application and system interactions affecting performance

What are we doing now?

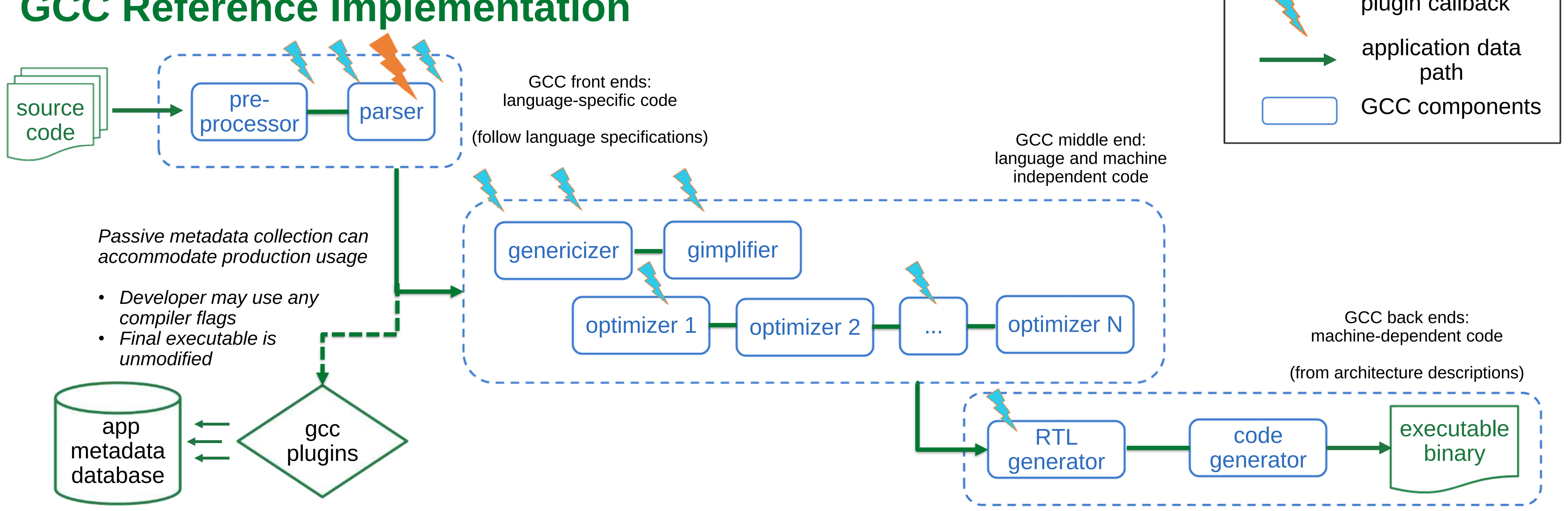
The community most often relies on ***single-use, labor-intensive*** efforts, ***"institutional knowledge",*** or written survey responses and ***anecdotal input*** from developers in an attempt to gain insight into diverse and expanding HPC applications.

Application	C	C++	F90	OpenMP	MPI	OpenACC	CUDA	CUDAF	Other	Libraries	Algorithms
DIRAC	x	x	x	x	x					BLAS, LAPACK, MKL, ALPS, HDF5	CCSD, tensor contraction
FLASH	x		x	x	x	x?	x			BLAS, LAPACK, HDF5, mesa	finite volume, stiff ODE solver, AMR
GTC	x			x	x		x			PETSc, Hypre, Adios	particle-in-cell
LSDALTON			x	x	x	x				BLAS, LAPACK	DEC Scheme, CCSD, RIMP2
NAMD		x					x		CHARM++	FFTW	MD, PME
NUCCOR			x	x	x					BLAS, LAPACK	CCSD, Hartree-Fock
QMCPACK		x		x	x		x			BLAS, LAPACK, FFTW, HDF5/Adios, libxml2	Monte Carlo
RAPTOR		x	x		x				kokkos, gampi (global arrays for mpi)	kokkos, HDF5	LES, finite volume, AMR
SPECFEM			x	x	x		x		OpenCL	Boast	Spectral Elements
ACME	x		x	x	x	x		x		trilinos, MOAB, PIO (NetCDF)	Spectral Elements, Voronoi tesselation, FCT
HACC		x		x	x		x		OpenCL, MPI-IO (custom)	FFTW, Thrust	P3M (particle-particle, particle-mesh) Tree PM
NWCHEM			x				x		Global Arrays (ARMCI)	PeIGS, ScaLAPACK	CCSD
XGC1	x		x	x		x		x		PETSc, LAPACK, PSPLINE, SuperLU, ParMETIS, CUDA Thrust	particle-in-cell

Applications on Titan in 2015



GCC Reference Implementation



What should we be doing?

→ Enabled through CAASCADE

Automate the collection of application program characteristics from a variety of tools.

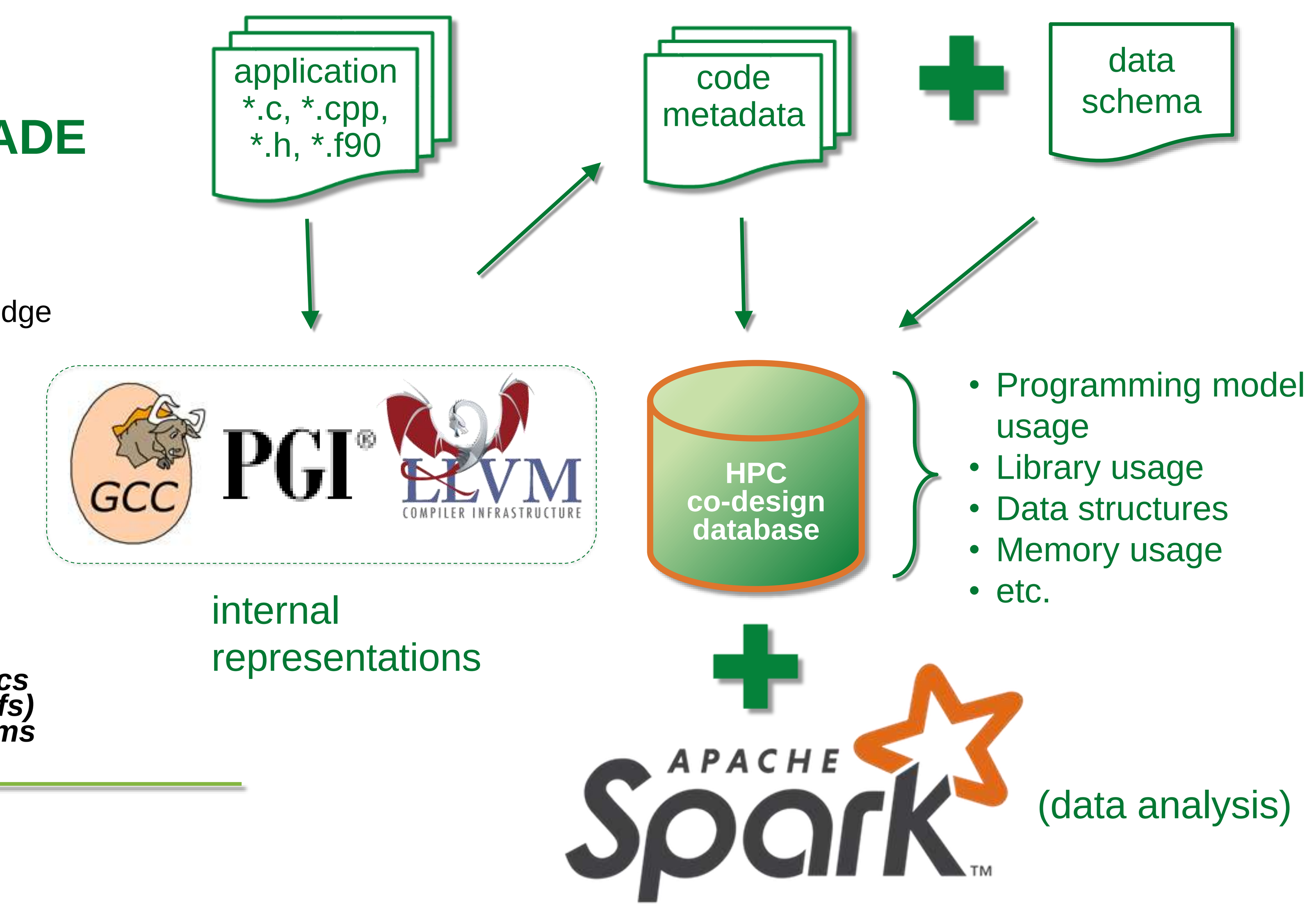
Create a database to ***provide access to this information,*** enabling data analytics and knowledge discovery techniques to ***inform ongoing HPC research.***

Ideal:

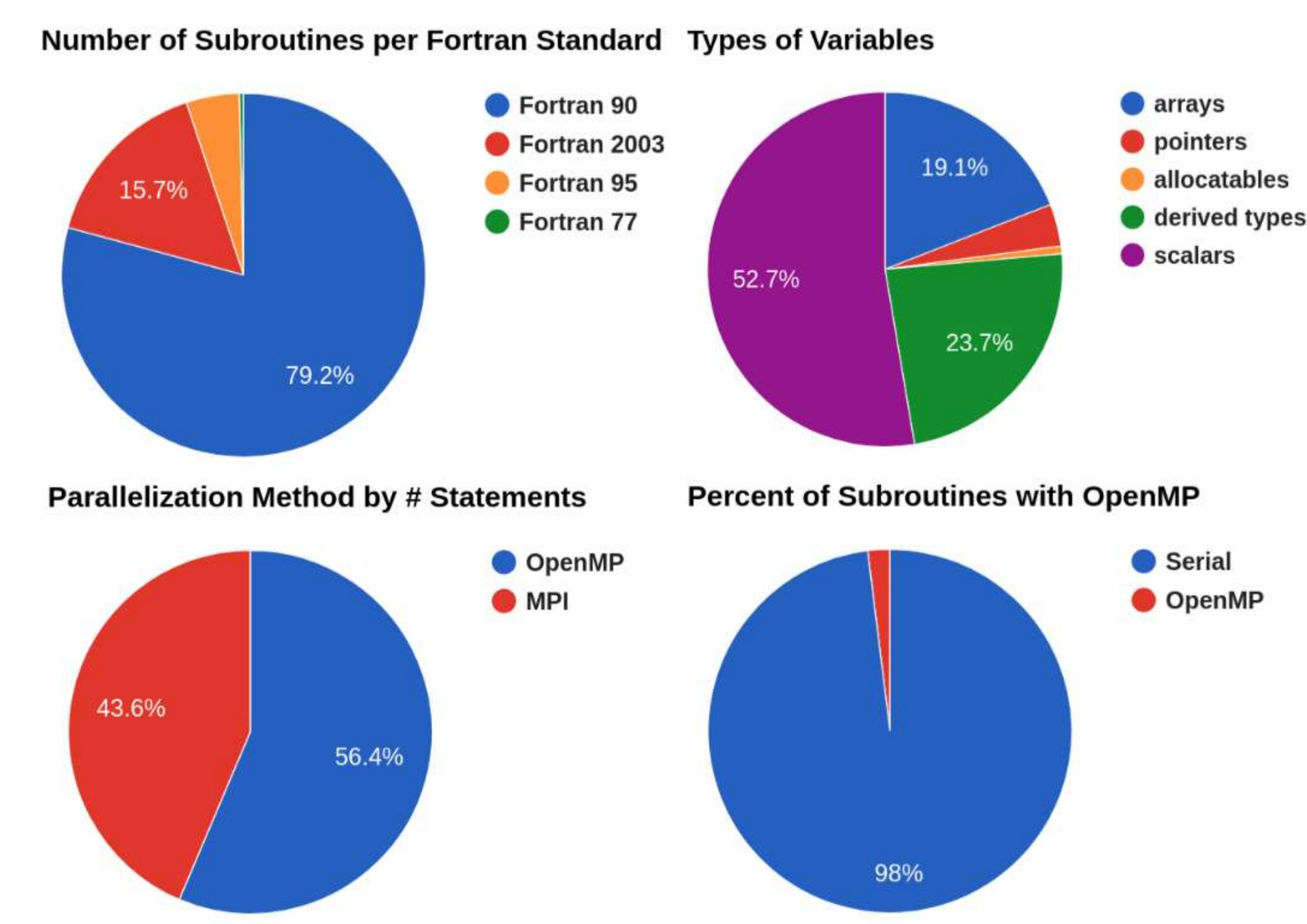
- Gather data in a repeatable, normalized (non-human) way
- Compilers and linkers know "everything" about source code

→convert compiler internal representations to human knowledge

→***understand which application characteristics (e.g language features, computational motifs) are contributing flops on Leadership systems***



Results

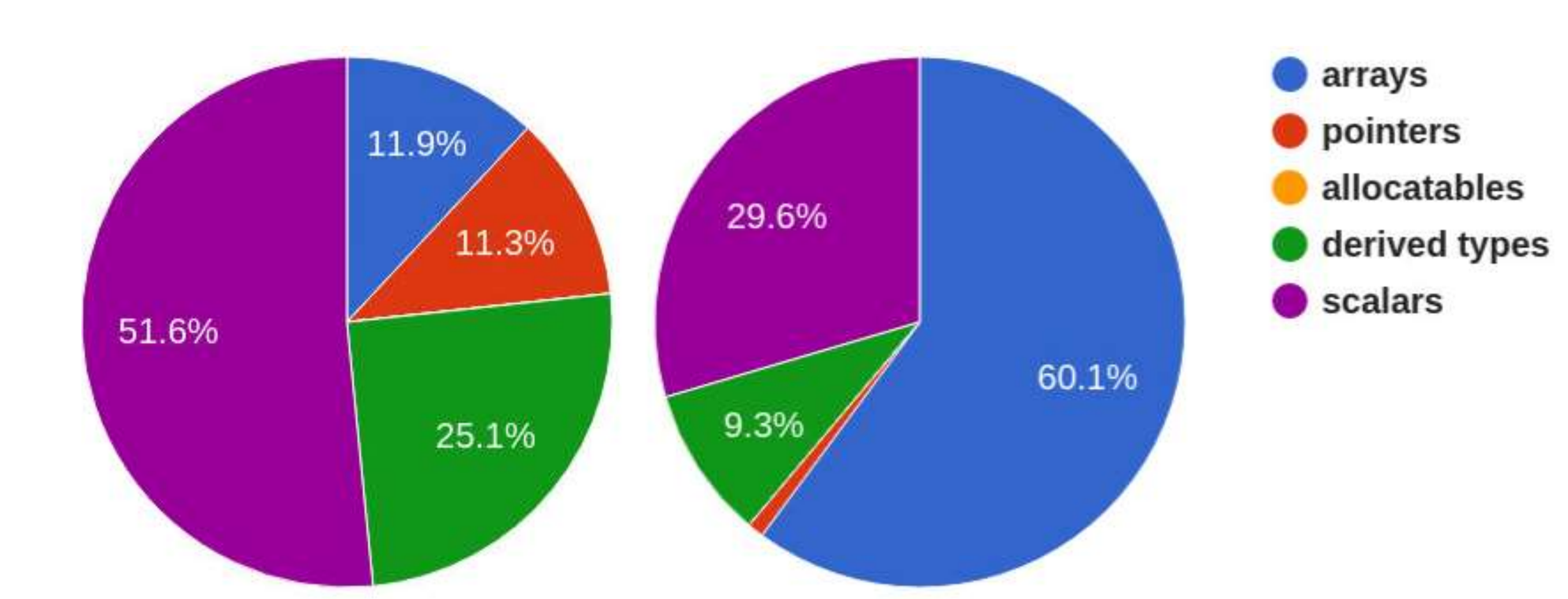


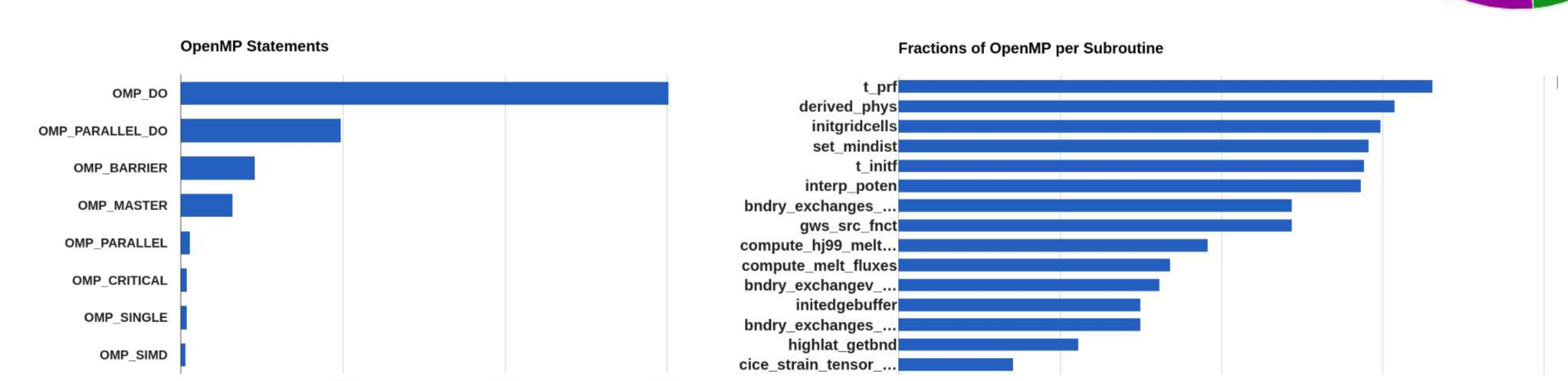
Left

High-level information such as Fortran language standard (top left), the type of variables (top right), OpenMP and MPI parallelization methods by the number of statements (bottom left), and subroutines with OpenMP pragmas (bottom right) in E3SM as collected by CAASCADE.

Below

The distribution of data type characteristics from QMCPACK from the CAASCADE static analysis only (left) and the same data when re-weighted by using dynamic runtime information from CrayPAT (right). The static only information indicates how the application is being developed and importance for programming strategies being used, while the dynamically-weighted information provides insights for performance considerations.



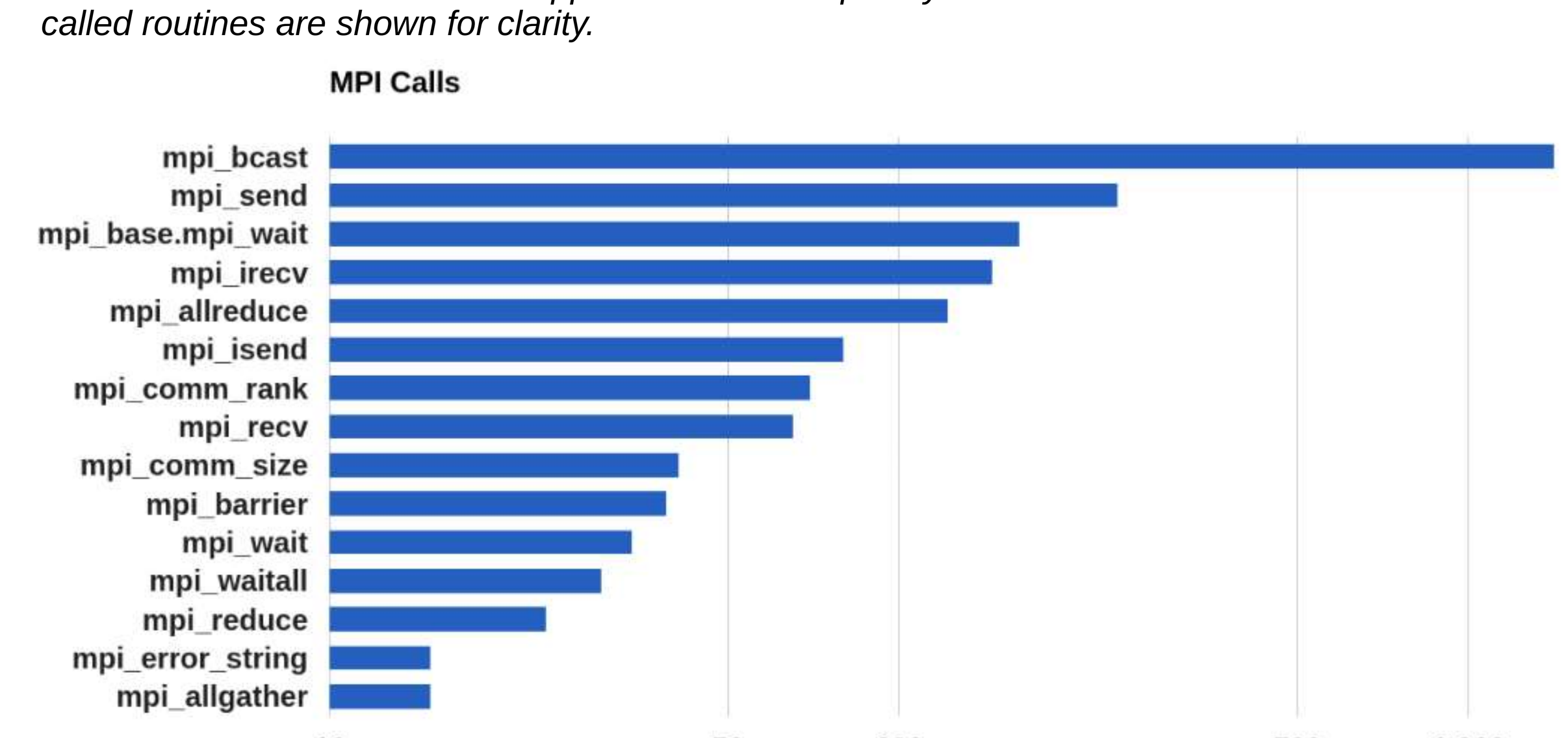


Above

Summary information about OpenMP usage in the E3SM application. The usage of specific OpenMP statements is shown (left), and the proportion of code covered within OpenMP lexical extents is shown per subroutine (right). Only the top fifteen are shown for clarity.

Below

MPI routines used in the E3SM application. The frequency of calls for each MPI routine is shown. Only the fifteen most called routines are shown for clarity.



Below

Libraries and their number of call sites detected with XALT and CAASCADE in NUCCOR. Libraries depicted in red belong to LAPACK and in blue belong to HDF5.