The need for software deployability: Broadening community tools for industry use

Benjamin Cowan

Tech-X Corporation

Overview

- Building a broad base of users and developers for community software is important for sustainability
- What is needed for industry to participate?
- From point of view of commercial HPC software developer

Tech-X History Overview

- Founded in 1994
- ~35 people, 2/3 Ph.D.s,
- Located in Boulder, Colorado, USA
- Leader of national projects, partner with national labs

Tech-X's mission is to provide customers with the best computational software and engineering services to enable their breakthroughs in research, development, design, and operations

TECH-X

Simulation Software Products

VSim	FDTD electromagnetics and kinetic plasma PIC code
USim	Shock capturing plasma fluid dynamics
PSim	Polymer physics modeling

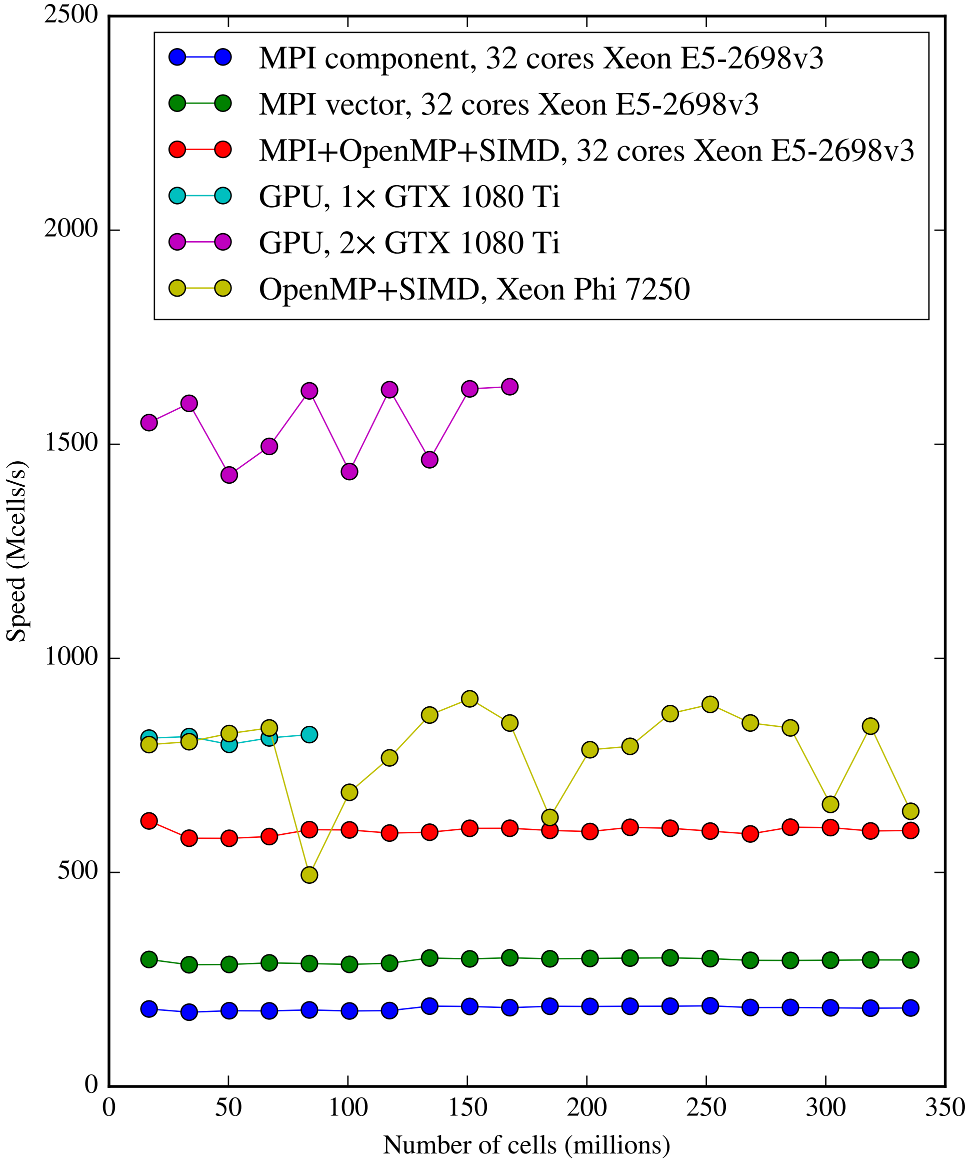
- Software provides unique physics and computational capability
- Works on multiple platforms, scales from laptops to supercomputers

VSim Application Areas

- Antennas
- Waveguides
- Microwave devices (e.g. klystrons, traveling-wave tubes)
- Magnetron sputtering
- RF-driven plasmas for semiconductor processing
- Optical fibers
- Silicon photonics
- Ion thrusters
- Plasma-based particle accelerators

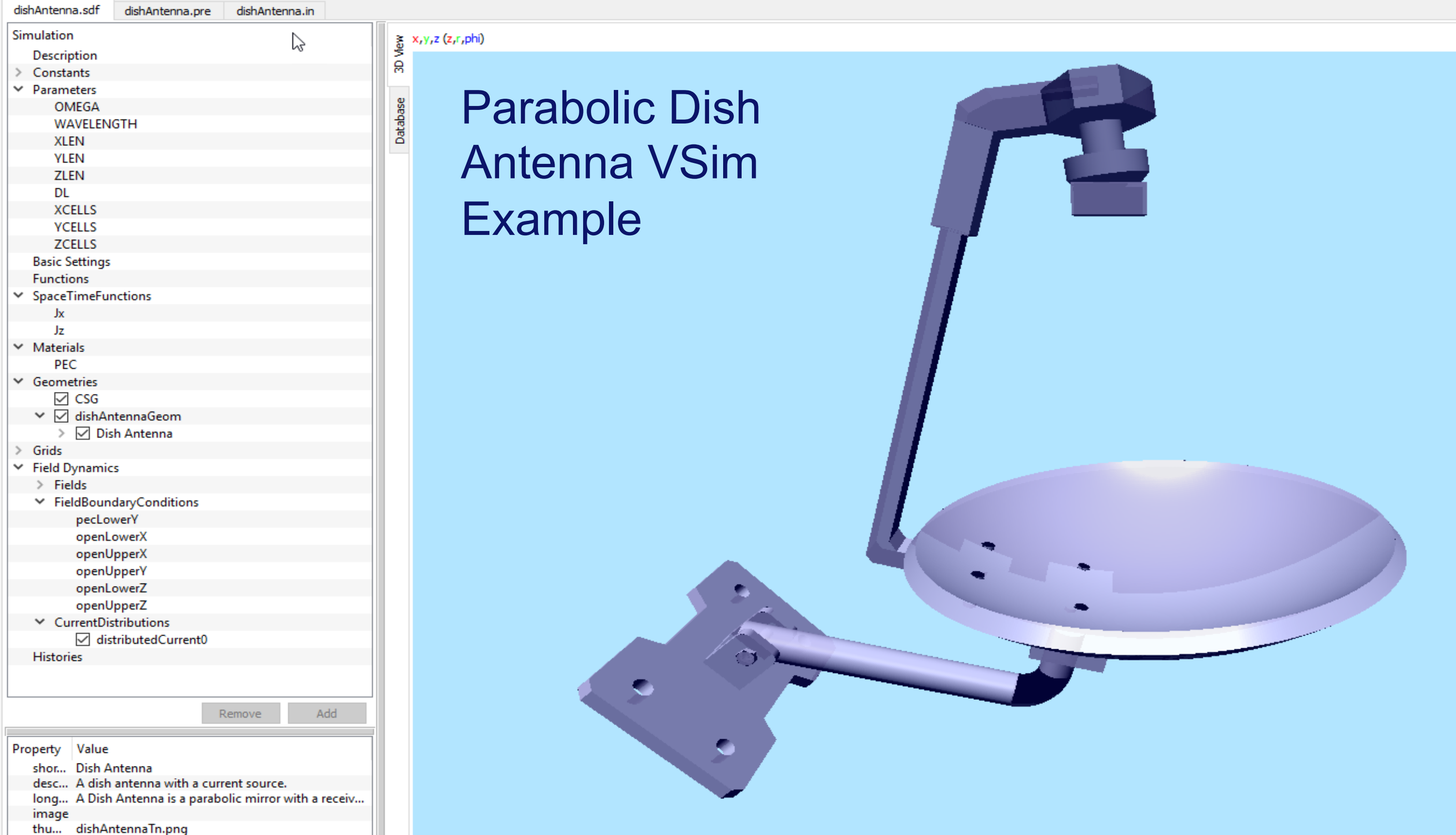
Performance portability

- Project underway since 2015 to bring high performance to VSim features on advanced architectures
- CUDA GPUs, thread parallelism and vector instructions on CPUs
- Puts additional constraints on how we use community software



Business trajectory

- Originally grew out of DOE SBIR program
- Increasing emphasis on commercialization over the last decade
- 2008: GUI development began; improved every version
- 2011: DOE SBIR program institutes commercialization requirements
- We target projects that lead to commercializable IP
- Closed-source model
- Sales steadily increasing; now support full time staff of application engineers



Property	Value
shor...	Dish Antenna
desc...	A dish antenna with a current source.
long...	A Dish Antenna is a parabolic mirror with a receiv...
image	
thu...	dishAntennaTn.png

Parabolic Dish Antenna VSim Example

Customer profile

Picture of Summit supercomputer at ORNL

Picture of desktop Windows box

Picture of broke Monopoly guy

Picture of Monopoly guys running with bags of cash

Deployability: Supporting commercial customers

- We can’t assume that the customer:
  - Can build software
  - Can install dependencies
  - Can manage drivers/system software
  - “I don’t have administrator privileges on my computer.” –Magnet engineer at national lab partner
- So we have to:
  - Provide installation in user space via installer or tarball
  - Have software perform well on customer machine without access to it
  - Support Windows

Constraints of deployability

- Can’t just use container or VM
- Need to build for compatibility with users’ expected software and drivers
- Testing is critical: Nightly, all platforms and hardware/compiler variants

Participation in community software

- We contribute to community software—if we can use it
- Trilinos: Linear algebra/solvers
  - Also SuperLU, HYPRE
- VisIt: Embedded visualization
- HDF5: I/O
- CMake: Build system generator
- Not adopted yet —no Windows support:
  - Kokkos: Performance portability. Got building on Windows with MSVC and LLVM, but atomics aren’t working
  - Spack: Package management

Windows

- How the Windows build environment looks, to an HPC developer:

Picture of dumpster fire

- Different shell environment
- Many packages require Visual Studio
  - Which lags in HPC features
- But catastrophes are preventable with preparation

Picture of flammable materials storage cabinet

CMake

Fat binaries

- Take advantage of vector instructions
- What vector instructions does our customer’s machine support?
- CUDA supports fat binaries and automatic dispatch for NVIDIA GPU architectures
- But host compilers are all over the map
- Compile flags? Attributes? Automatic dispatch?
- Maybe just build shared libraries for each architecture; resolve at install or load time. Infrastructure required?

The “rainbow of doom”

Feature	Linux		Mac				Windows			
	GCC	Intel	Apple Clang	LLVM Clang	GCC	Intel	VS 2017	Clang	Intel	MinGW w64
CUDA	✓	✓[3]	✓	?		x[3]	✓	?[1]	x[3]	x
OpenMP	✓	✓	x[2]	✓[2]		✓	x	✓	✓	✓
Kokkos works	✓	✓	✓	✓		✓	x[4]	x[4]	✓	✓
target clones	✓	✓	x	x	x	✓	x	x	✓	x
function multi-versioning	✓	✓	x	x	x	✓	x	x	✓	x
target attribute	✓	?	✓	✓		?	x	✓	✓	✓
Builds engine toolchain	✓	✓	✓	✓		✓	✓	✓	✓	✓
Performance	B	A	B	B		A	?	?	A	
Cost	Free	💰💰💰	Free			💰💰💰	💰	Free	💰💰💰	

1. https://llvm.org/docs/CompileCudaWithLLVM.html: CUDA compilation is supported on Linux, on MacOS as of 2016-11-18, and on Windows as of 2017-01-05, but failing in trunk (https://bugs.llvm.org/show_bug.cgi?id=38811) for Windows.
2. https://openmp.llvm.org/. Apple’s clang does not contain openmp. Download LLVM7 or use llvmall to build.
3. https://docs.nvidia.com/cuda/cuda-toolkit-release-notes/index.html: CUDA 10.0 supports ICC-18, but on Linux only.
4. Kokkos builds, but tests failing. Clang requires LLVM kludge for long pathnames.

Recommendations

- Support Windows
  - It’s easier if you start early
  - Encapsulate Windows-specific issues
- Can your software be deployed without the user having to build it?
- Thoughts on open source:
  - Our code encapsulates our competitive advantage—in both commercial and research sectors
  - Our target users are not software developers
  - Avoid GPL—Software that’s “free as in speech” is a nice ideal, but beer costs money