Emerging Technologies in Productivity for HPC

Rob Hoekstra

PSAAP III Pre-Proposal Conference, March 14, 2018
Complexity UP, Productivity DOWN

- HW is more complex
- Software stack is more complex
- Programming Environment/Model is more complex
- Execution/Operations Environment is more complex
- All these factors can negatively impact PRODUCTIVITY
Productivity has been declining rapidly in the HPC environment

- Dramatic increase in complexity of algorithms and applications coupled with a dramatic increase in complexity, diversity and scale of HW and execution environments

- AND CS/CSE research on productivity pays little attention to our HPC-specific problems (there are counter-examples such as IDEAS)
Even worse for our Mission Codes

- Complexity, size and dependencies of our codes is well above average even in the HPC community

- Verification/validation requirements create a much higher bar for incorporation of new capability whether it be physics, algorithms or performance optimization

- And to make it worse, leadership-class platforms environments (SW stack, etc.) are often be more immature/fragile than average
Bottlenecks

- Code development
- Code correctness/testing
- Platform specific tuning/optimization
- Problem setup
- Job Execution & Steering
- Analysis & Viz
Code Development

- MPI/Fortran code

- C++, hierarchical parallel constructs, layered dependencies
Testing/Verification

- “Eyeball” Norm

- Large verification test suites, non-reproducibility, etc.
Performance tuning/optimization

- PRINTF (still fall back to this many times:)

- Performance analysis and “divination”

```
Frame Domain / Frame / Function / Call Stack | CPU Time
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>[No frame domain - Outside any frame]</td>
<td>19227.430s</td>
</tr>
<tr>
<td>Nalu: AssembleMomentumElemSolver</td>
<td>15636.292s</td>
</tr>
<tr>
<td>Nalu: TpetraLinearSystem: finalizeLinearSystemA</td>
<td>13508.756s</td>
</tr>
<tr>
<td>N2KokkosSparse4mplt2SPMV: Function: k0: Matrix: k0: k0</td>
<td>12450.998s</td>
</tr>
<tr>
<td>N2KokkosSparse4mplt2SPMV: Function: k0: Matrix: k0: k0</td>
<td>18026.345s</td>
</tr>
<tr>
<td>Nalu: turbViskksAlgorithm: execute</td>
<td>9090.105s</td>
</tr>
<tr>
<td>Nalu: AssembleScalarElemSolverAlgorithm: execute</td>
<td>6697.490s</td>
</tr>
<tr>
<td>N18KokkosBlas4mplt1MV_Update: Function: k0: k0: k0: k0: View: k0: k0: k0: k0</td>
<td>4215.220s</td>
</tr>
<tr>
<td>Nalu: AssembleContinuityElemSolverAlgorithm: execute</td>
<td>3705.461s</td>
</tr>
<tr>
<td>N2KokkosSparse4mplt2SPMV: Transpose: Function: k0: Matrix: k0: k0</td>
<td>2587.633s</td>
</tr>
<tr>
<td>N2KokkosSparse4mplt2SPMV: ViewDefaultConstruct: k0: ViewDefaultConstruct: k0: k0: k0</td>
<td>2431.859s</td>
</tr>
<tr>
<td>Nalu: TpetraLinearSystem: buildElemToNodeGraph</td>
<td>2297.399s</td>
</tr>
<tr>
<td>Nalu: AssembleNodalGradElemAlgorithm: execute</td>
<td>1848.291s</td>
</tr>
<tr>
<td>Nalu: ComputeEMdotElemAlgorithm: execute</td>
<td>1835.931s</td>
</tr>
<tr>
<td>AssembleNodalGradElemAlgorithm: execute</td>
<td>1512.716s</td>
</tr>
</tbody>
</table>
```

- Effective Time by Utilization:
  - Ti
  - Phi
  - Phi
  - Phi
  - Phi

- Instructions Retired
  - Phi
  - Phi

- CPI Rate
  - Phi
  - Phi
  - Phi
Problem Setup

- Card Deck

- Complex workflow with geometry/meshing, etc.
Cubit Hex Meshing Capability

housing geometry has 13 ‘volumes’
decomposed into meshable volumes

Cubit Journal file – 6200 lines long
Manually constructed
800+ manually specified webcuts defined
1500+ geometry cleanup commands
500+ meshing commands
13 volumes to 500 webcut volumes
1000+ hours of tedium

Turn around time:
9 months
Job execution/steering

- C:> Run app.exe

- Complex workflows of multi-physics, multiple codes, steering, data collection
Analysis/Viz

- Quantity = X

- Complex data flows/viz packages/UQ/validation
Areas of opportunity

- What are future HPC “High Productivity” Programming Models?
- What are future HPC “High Productivity” Development Environments?
- What are future HPC “High Productivity” Runtime/Execution Environments?

AND

- Is there a more coherent unification of design time, compile time and runtime environments/tools?
Programming Models

- What are future HPC “High Productivity” Programming Models?
  - Portability Abstractions
  - Async Multi-Tasking
  - DSLs
  - Component-based development

Legion
A Data-Centric Parallel Programming System

DARMA
Uintah
Charm++
RAJA

κόκκος
kokkos / grain; scarlet; seed
Development Environment

What are future HPC “High Productivity” Development Environments?

- IDEs
- Auto-tuning
- Higher-level languages/scripting
- Open compiler environments
- Automated testing
- CSE SW Engineering “Best Practices”
IDEAS: Interoperable Design of Extreme-scale Application Software

- Project began in Sept 2014 as ASCR/Ber partnership to improve application software productivity, quality, and sustainability

Resources: [https://ideas-productivity.org/resources](https://ideas-productivity.org/resources), featuring

- WhatIs and HowTo docs: concise characterizations & best practices
  - What is Software Configuration?
  - What is CSE Software Testing?
  - What is Good Documentation?
  - How to Configure Software
  - What is Version Control?
  - How to Write Good Documentation
  - How to Add and Improve Testing in a CSE Software Project
  - How to do Version Control with Git in your CSE Project
  …. More under development

PIs: Michael Heroux (SNL) and Lois Curfman McInnes (ANL)

Co-PIs: David Bernholdt (ORNL), Todd Gamblin (LLNL), Osni Marques (LBNL), David Moulton (LANL), Boyana Norris (Univ of Oregon)
Runtime/Execution Environment

- What are future HPC “High Productivity” Runtime/Execution Environments?
  - Workflows
  - Tasking
  - Machine Learning
  - Problem Setup
  - Containers
Productivity improvement can be a common thread in PSAAP center activities

- "Focus" on productivity enhancing technologies that are highly synergistic with other goals
  - Workflows
  - Programming Models/Environments
  - Machine Learning
  - Component-based Approaches

- Tell us how your center will leverage research in these areas will have a big positive impact on PRODUCTIVITY.
Questions?
Visible Progress (Writing code, computing results)

Planning

Recoding and Porting to new Platforms

Adapted from Software Project Survival Guide, Steve McConnell