MS65: Chapel Meets Serious Applications – Evaluating a High Productivity Language

Jonathan Claridge, University of Washington

Jonathan Turner, University of Colorado

John Lewis, Cray Inc.

March 2, 2011

Introduction

- Chapel – the Cascade High Productivity Language
- DARPA 2002 call for High Productivity Computer Systems (HPCS)
  - DARPA's Vision – create peta-scale computer systems *that were much easier to program*
  - Part of Cray's response – develop a new language designed for large scale parallel systems

Organization

1. Chapel – What is it? What makes it better?

   Jonathan Turner, An Overview of Chapel

2. Does Chapel really make us more productive? Evaluating Chapel with Programming Motifs

   - John Lewis, Linear Algebra Programming Motifs
   - Jonathan Claridge, Adaptive Mesh Refinement in Chapel -- Part I: “Simple” problems with nice solutions
   - *Jonathan* Claridge, Adaptive Mesh Refinement in Chapel -- Part II: The Hard Part

Evaluating a Programming Language

- Demonstrations with only toy problems neither tests a language nor convinces potential users
- Hard to write complete applications in a new language
- A middle way – demonstrate language capabilities on moderate-sized problems that illustrate the complexity of real applications
- Challenge – find a set of computations that span the breadth of programming idioms / motifs of real applications

Programming Motifs

Berkeley "Thirteen Programming Motifs":

http://www.eecs.berkeley.edu/Pubs/TechRpts/2006/EECS-2006-183.pdf

- Identified classes of computations / algorithms, representing *the most important computation and communication patterns*
- Abstract – deliberately not benchmarks, not code

Mini-applications

Chapel Evaluation Plan:

- Ten motifs relevant to supercomputing
- Specific instances (mini-applications) of motifs that exercise Chapel features
- Write Chapel programs
- Where possible, compare to existing codes in language-neutral manner
  - For instance, compare length in tokens – symbols in language
    - Variable names
    - Reserved words
    - Reserved symbols

Selected Mini-Applications

Motif / Application Area		Candidate Codes
1	Dense Linear Algebra	HPCC HPL benchmark
2	Sparse Linear Algebra	Sparse Multifrontal or SuperLU Factorization
3	Spectral Methods (FFT)	HPCC G-FFT
4	N Body Problem	Boeing DARPA Fast Multipole Method
5	Structured Grids	UW AMR Framework
6	Unstructured Grids	Mantevo phdMesh or miniFE
7	Embarrassingly Parallel	NAS EP or SSCA #2 Kernel 4
9	Graph Algorithms	SSCA #2
10	Dynamic Programming	SSCA #1 or Smith-Waterman
14	Parallel Input / Output	SSCA #3

Chapel is freely available….

The compiler and all the codes referenced today are open source. View

http://chapel.cray.com/

for copies of these presentations and links to these codes.