Final Defense Practice

Andrey Asadchev

VT.edu

February 8, 2013

About Me

You know me already

Previous Research

- GAMESS (*vomit)
- GPU (*vomit)
- Rys Quadrature: C++ with Mathematica optimized, Python generated expressions
- Hartree-Fock
- MP2: can handle large systems, few thousand basis
- CCSD(T): runs on large and small machines, good I/O, disk as fallback

Interests

- C++
- Python
- Beautiful code and Domain specific languages
- Hardware-level optimization and Parallel computing

CI - current project

C++

My evolution: F77, F90, C, C++

General purpose language, aged well (C++11)

Templates and preprocessor

Domain specific language - eg Eigen

Access to hardware instructions

Not an easy language to master

Python

Very versatile, rapid prototyping language

Interacts with C/C++

Plethora of math and graphics packages

Mathematica-like shell for Quantum

Beautiful code

Code as expression of oneself, a bit artistic pursuit

Concise code lends itself to comprehension and easier manipulation

Easy manipulation leads to better algorithm

Domain specific languages (C++ or Python) allow expressing formulas concisely

Lower bar for beginners and more power to advanced folk.

Optimization

Few tricks: blocking, loop unrolling

Templates help a lot

Requires a little background in computer arch.

Parallel computing is usually data-driven in my approach (data storage determines the algorithm)

Some construct to transform loop to parallel would be nice (Rose)

In the end, *my* idea is to parallelize on algorithm level (rapid prototyping is handy)

Which leads to C++/Python alliance: C++ handles heavy lifting, Python ties the things together

CI

Large matrices (that need be stored for several cycles)

Some fun bit manipulation

Optimized to hide I/O (blocking)

Does ok atm, but needs more optimization

(16,16) Full CI in 6-40 mins (single thread)