Fortran Background and Introduction

Shirley Moore
CPS5401 Fall 2014
http://svmoore.pbworks.com/
September 24, 2014

Fortran

- Stands for “Formula Translation”
- Originally developed in the 1950s by a team led by John Backus at IBM at their campus in south San Jose, California
- Intended for scientific and engineering applications
- One of the first high-level languages
  - Freed programmers from writing in machine-specific assembler language
  - Programs were (somewhat) portable

Fortran for Scientific Programming

- One of the most popular languages in the area of high-performance computing
- In continual use for over half a century in computationally intensive areas such as numerical weather prediction, finite element analysis, computational fluid dynamics, computational physics and computational chemistry
- The language used for programs that benchmark and rank the world's fastest supercomputers

Fortran Standards

- Fortran 66 – ANSI standard consisting of a common subset of existing dialects
  - But most compilers did not adhere to the standard
- Fortran 77 – ANSI standard based on vendor extensions and preprocessors
- Fortran 90
  - Responded to development in language design
  - New features: array operations, pointers, user-defined derived data types, modules for encapsulation, new control constructs, dynamic storage, recursion
  - Fortran 77 retained as a subset
- Subsequent standards
  - Fortran 95 – minor revision
  - Fortran 2003 – major revision
  - Fortran 2008 – minor revision

Fortran vs. C/C++ for Scientific Programming

- Problem dependence
  - Fortran excels at array processing. If your problem can be described in terms of simple data structures and particular arrays, Fortran is well suited.
  - Fortran is better for *numeric* scientific computing.
    - finite differences/elements, PDE solvers, electronic structure calculations
  - C++ is better suited for complex and highly dynamic data structures.
    - graphs, mesh generators, symbolic manipulation
- Skill dependence
  - It takes a lot more programming experience to write efficient C/C++ programs than to write efficient Fortran programs.
  - You will probably get a better return on investment learning Fortran than learning C/C++, assuming that your problem is suited to Fortran.
    - Easier for a scientist to write *fast* programs in Fortran than in C/C++
- Project dependence
  - The people you are working with
  - Legacy code
- Possible to combine them

Resources

- Fortran 90 Tutorials
  - http://www.cs.mtu.edu/~shene/COURSES/cs201/NOTES/fortran.html
  - http://www.owlnet.rice.edu/~ceng303/manuals/fortran/
- Gfortran – the GNU Fortran compiler
  - http://gcc.gnu.org/wiki/GFortran/