Is Fortran getting further and further away from the Hardware Architecture?

John Levesque
Director
Cray Supercomputing Center of Excellence
Future hardware directions

• Flop rich environment
  – More flops per clock
  – More cores per socket
  – More threads per core
• The memory wall is thicker and thicker
  – Increased importance of cache re-use
  – Effective use of hardware and software pre-fetching
Future Compiler/User Concerns

• Flop rich environment
  – More flops per clock
  – More cores per socket

• Compilers must vectorize more code, no longer only for Cray and NEC.
  – New CONCURRENT directive will help

• Compiler/Users must employ threads across the cores to mitigate memory bandwidth issues
  – OpenMP and shared caches?
  – Pthreads to hide memory latency
Future Compiler/User Concerns

• Increased importance of cache re-use

• Users must take care in blocking loops to more effectively utilize multi-levels of cache
  – Not trivial
  – May not be portable
    • Different size caches
Future Compiler/User Concerns

• Memory Wall

• Users must avoid striding, storing when not required,.....
  – Never, never use an array when a scalar will do
  – Don’t pass array sections
Fortran features that ignore hardware issues and performance

- Derived Types
  - Inhibit many compilers’ ability to vectorize DO loops
  - Can help cache reuse
- Modules
  - Inhibit Compiler and User from scoping variables. EG scoping a data element from a module as private. Same problem with COMMON blocks
  - Sub-modules may help
- Array section passing
  - Incurs increased data motion to and from memory at the subroutine boundary
- Array assignments
  - Implies that the operation be performed for the full extent of the array limits
    - Very Cache unfriendly
    - Takes a good compiler to solve this
What Can Be done in the language?

• DONTNEEDTOSTORE directive.
• Advisories to the users as to the performance pitfalls when using certain Fortran statements
• PLEASE, PLEASE, don’t say it is up to the compiler.
  – There are some compilers that can deal with the issues; however, not all compiler can deal with the issues.
    • Performance Portability goes out the window