A Compiler Engineer’s View of High Performance Technical Computing

Instruction Set Architectures in Top 500

PGI* Compilers & Tools
Architectural Diversity

Instruction Set Architectures in Top 500

Architectures
Architectures (>1)
Architectural Diversity

Instruction Set Architectures in Top 500

- Red: Architectures
- Blue: Architectures (>1)
Architectural Trends

- Vectors getting longer
- Multiprocessors getting larger
- Memory hierarchies getting deeper and more interesting
- Heterogeneity becoming common
Language Trends

- Fortran evolving; array operations, do concurrent, PGAS parallelism
- C and C++ each looking to add loop and task parallelism
- OpenACC parallelism across accelerators and multicore
- OpenMP task parallelism, simd parallelism, device constructs
metalanguages

do 10 i = 1, len
 10     y(i) = y(i) + a*x(i)
subroutine daxpy( y, a, x, n )
    real y(*), x(*), a
    integer len, i
    do 10 i = 1, n
10       y(i) = y(i) + a*x(i)
end

....
call daxpy( y, a, x, len )
void daxpy( float* y, double a, float* x, int n ){
    for( int i=0; i<n; ++i )
        y[i] += a*x[i] ;
}

... 

daxpy( y, a, x, len ); saxpy(); caxpy();...
forall( 0, len, [&](int i){
    y[i] += a*x[i] ;
} );
Compiler Challenges

- Effective parallelism management and exploitation
- Predictable performance (no surprises)
- Data management (cache, scratchpad, HBM)
- Heterogeneity (ISA, performance)
- More parallelism in language (Fortran, more to come)
- C++ keeps getting more interesting
- PEACE continues in force