Characteristics of PM/PE(s) (set 1)

Mary Hall
University of Utah

What Tools Comprise the PE?

From workshop materials:

- Compilers
- Code transformation
- Code synthesis
- Code generation
- Debuggers
- Autotuning
- Runtime systems
- Workflow management
- Data analytics
- Visualization
- Storage management

Added during discussion:

- Optimized libraries
- Performance analysis and visualization
- Performance modeling
- Correctness tools
- Resilience tools
- Software quality tools
- Build-and-test frameworks
- Virtualized testbeds

Performance and Correctness at Scale

- Performance monitoring and analysis
  - More sensors throughout hardware/software stack, sophisticated filters, longer term vs. instantaneous measurements
  - Includes power/energy management
- Performance and correctness debugging
  - Attribution to cause (h/w, s/w, multi-physics, etc.), in light of dynamic adaptation
- Verification for high thread counts
  - Static and dynamic
- Introspection as part of application
  - Expert programmer builds performance expectations into application for self diagnosis
- Coverage testing (correction/performance) for execution environment design space
  - In light of dynamic application build and execute and resource availability
  - e.g., Fault injection as part of testing process(?)
- Support for verifiability of scientific results
  - In light of bit flips, relaxed dependences, new communication-avoiding algorithms, etc.
- Data analytics and visualization
  - Similar concerns to performance and correctness debugging (in situ)

Performance Portability

- Domain-specific optimizations/languages/libraries
  - Underlying architecture-aware mapping for high performance
- Autotuning
  - Currently baked into application through autotuning library, or result of offline tool (a la SUPER, TUNE)
  - Expand: Compiler-directed or express code variants and optimization parameters at application level
  - Integrate into application build process
- Mid-level mapping tools (TiDA/HTA/Raja/Kokkos)
- Dynamic code generation, compilation
  - Opportunity to specialize for execution context
- Run-time optimization

How Dynamic Is Application Build and Execution?*

- Dynamic code generation and JIT compilation
  - Opportunity for architecture and input data specialization
  - Perhaps execution environment, too
- Autotuning
  - Assemble code variants and adjust optimization parameter values at run time
- Asynchronous Many-Task
  - Incorporate into application specification
- Tolerance for non-deterministic execution? Testing?

*This list should not necessarily be considered an endorsement by the panel.