A Unified Programming Model for Multicore CPUs and Many-Core Accelerators

Michael McCool

November 18, 2008
SC08, Austin TX

RAPIDMIND

High Productivity Computing

Performance

- Fully leverage the potential of all processors/cores
- Optimize use of memory bandwidth, hide latency

Portability

- Support hardware independent applications
- Automatic scalability to multiple parallelism mechanisms
- Exploit a variety of multi-core processors efficiently
- Use many-core accelerators when available

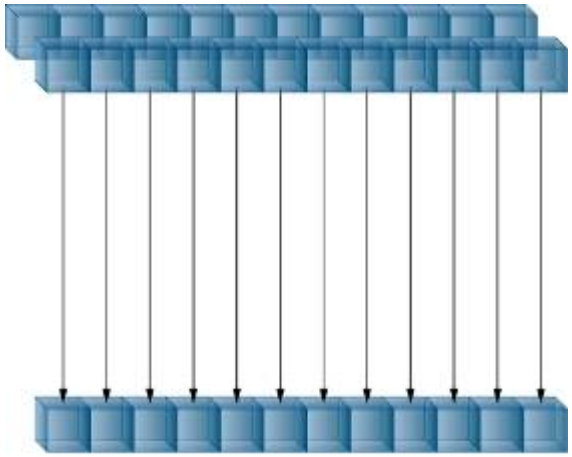
Programmability

- Simple extensions of existing practice
- Automate trivia, provide drill-down when needed
- Explicit control of important policy decisions

Structured Parallel Programming

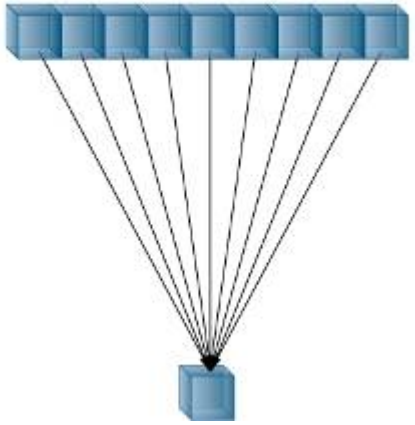
Apply functions to arrays:

- Application: `C = f(A,B)`
- May have control flow
- May perform random access into other arrays
- Can read and write to subarrays
- May have local arrays and function calls



Apply collective operations to arrays:

- Reduce: `a = reduce(p,A)`
- Gather: `A = B[U]`
- Scatter: `A[U] = B`
- Others: `scan, pack, combine, …`



Use *structured* compositions of *deterministic* parallel *patterns*

Performance Benchmarks