Programming Models & Portability

Bill Carlson

Guang Gao

Emerging Features

- Do we need emerging programming models to support emerging architectures and applications?
  - Some features may not actually emerge!
  - Some features are trend-setters!
- In the end, can consider adding a few

Trend Setters

- Both XMT and Cell contribute trend-setting features
  - Importance of Locality (Cell)
  - Threading and Parallelism (Both)
  - Enhanced Synchronization (XMT)
- Also, want programming models to support more emerging architectures!

Locality

- Memory systems becoming hierarchical
- Currently specify domain decomposition
- Future specify hierarchy levels *explicitly*
  - *e.g.* what stays in local state for Cell
- Some think that such “vertical” level spec is more important than “horizontal” distance.

Parallelism

- Evolving programming model concept to execution model
- Threading will be critical: a thread is not a “processor”
- An execution model includes thread model, memory model and synchronization model.

Synchronization

- Need a toolbox of advanced concepts
  - Locks
  - Transactions/Atomics to avoid synchronization
  - Higher level algorithmic concepts: queues, trees, etc.
  - Consistency model control

How do we make this happen?

- Separation of concerns
  - Function, Parallelism, Locality, Synchronization
- Evolution from current models
  - Modify/Annotate existing code
  - Ensure starting point allows existing sequential code to be easily ported
- Research access to emerging systems
- Application development needs support to help emerging architectures and programming models