Architectural Scalability

General Comments

- People don’t parallelize in ways that don’t parallelize well on today’s machines
- Naming of processing resources should not be an integral part of exec/prog models
- Bad – chunk up memory requires more pools and size of pools esp. when you go to high concurrency
- Transactional Memory is for programmability not necessarily a scalability or performance issue
- To what extent is power consideration a part of scalability
  - Should be
  - Drives heterogeneity in today’s design
  - Today we’re sloppy about things

Keys to Scalability

- Key to scalability: Reduces Amdahl's law effects
- Key is fine scale granularity for create threads, comm., and sync
  - M-machine C threads shared registers – low thread creation costs
- This includes anything that generalizes to associative operations
  - inc. memory allocation
  - Importance of in-memory ops – want to do associative ops
  - Do it "in network" with aggregation
  - Then you can combine requests without performance loss
  - Generalize parallel prefix for massive serialization ops in run-time such as mem alloc

Cell/XMT Comments

- XMT is step in direction of anonomizing processing core
- 100 core Cell may be more scalable than 100 x86 core chip with h/w controlled caches
- Scalability: as you scale you need finer grain and Cell doesn’t work if you can’t pipeline it