Friar Tuck’s Chapel
Qthreads & the Forest of Thieves

Kyle Wheeler
Dylan Stark

SC11 Chapel BoF
Lightning Talk
Qthreads Highlights

- Lightweight User-level Threading (Tasking)
- Platform portability
  - IA32/64, AMD64, PPC32/64, SparcV9, SST, Tilera
  - Linux, BSD, Solaris, MacOSX, Cygwin
- Locality awareness
  - “Shepherd” as thread mobility domain & locality
- Fine-grained synchronization semantics
  - Full/Empty Bits (64-bit & 60-bit)
  - Mutexes
  - Atomic operations (Integer Incr, Float Incr, & CAS)
- Locality-aware Workstealing Scheduler Model: Sherwood
- Supports multiple programming models
  - Chapel
  - OpenMP
Chapel on Qthreads

• Chapel Tasking Layer Interface
  – Task management
  – Synchronization
  – Thin translation layer to Qthreads

• Implementation Details
  – Qthread environment bolted on the side
  – Spawns tasks from separate GASNet progress thread
  – Nightly testing by Chapel team

• Compiling with Qthreads is Easy!
  – set `CHPL_TASKS=qthreads` when building Chapel
Multi-Node Chapel on Multiple Qthreads

• Communication (via GASNet)
  – Blocking system calls
    • Dedicated OS thread
    • Solutions:
      – Forked initialization thread
      – Explicit progress thread creation
  – External Task Operations
    • Task creation from outside the task library
      – Memory management issue
      – Also: synchronization issue…
    • Task synchronization outside the task library
      – Proxy-task using thread-level synchronization (pthread_mutex_t)
Sherwood Scheduler

- **Basic idea:** combine workstealing and PDFS
- **Intra-chip shared LIFO queue**
  - Exploits shared L2/L3 cache
  - Natural load balancing across local cores
- **FIFO work-stealing between caches**
  - Maintains L3 cache locality
  - Balances load
- **Important details:**
  - Only one thief per chip performs work-stealing (avoid unnecessary communication)
  - Thief must steal multiple tasks, preferably enough for all cores sharing the on-chip queue
  - Not all tasks are stealable
Coming to a Chapel compiler near you...

- Better mapping from Chapel sync to Qthreads’ FEBs
- Better I/O handling
- Studies
  - Further benchmark studies
  - Workstealing/loadbalancing studies
- Hierarchical locales
- Task Teams
- Eurekas and other collectives
- Integrated inter-locale communication layer
Thank You!