MassiveThreads Tasking Layer

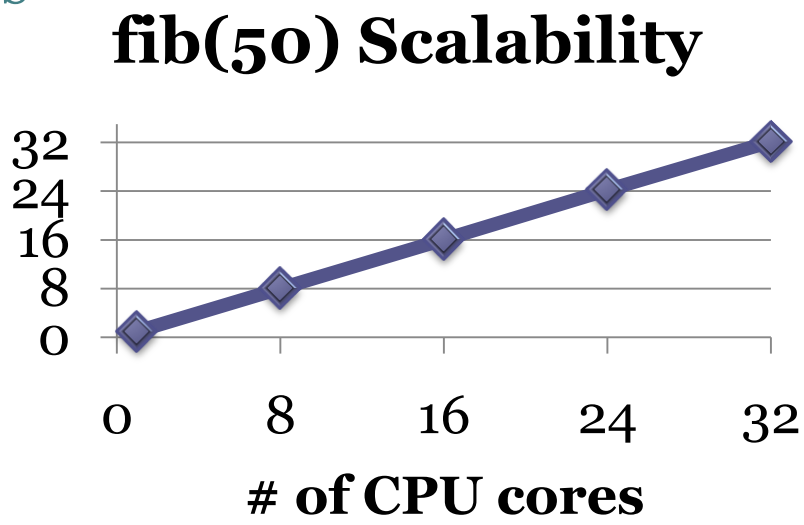
Jun Nakashima, Nan Dun and Kenjiro Taura
The University of Tokyo

What is MassiveThreads?

- A user-level lightweight thread library
  - Designed for High Productivity Languages
- Key characteristics:
  - Efficiently support fine-grain threads
  - Provide pthread-compatible API *and semantics*
    - Blocking I/O can trigger context switches
- We implemented Chapel tasking layer by MassiveThreads

Why MassiveThreads?

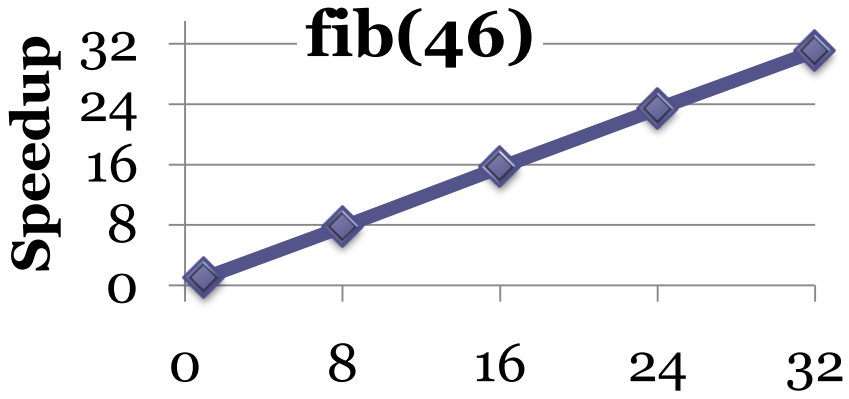
- Good performance on task-parallel application
  - Create-and-destroy: < 80ns
  - Scalable dynamic load balancing
- Support multiple locales
  - Can handle multiple I/Os concurrently
- Easy to integrate
  - Pthread-compatible API makes interaction between communication threads and tasks straightforward

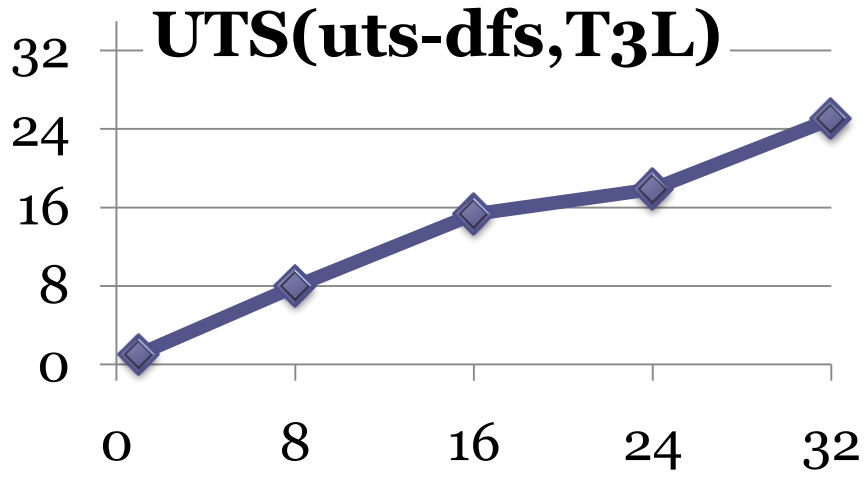


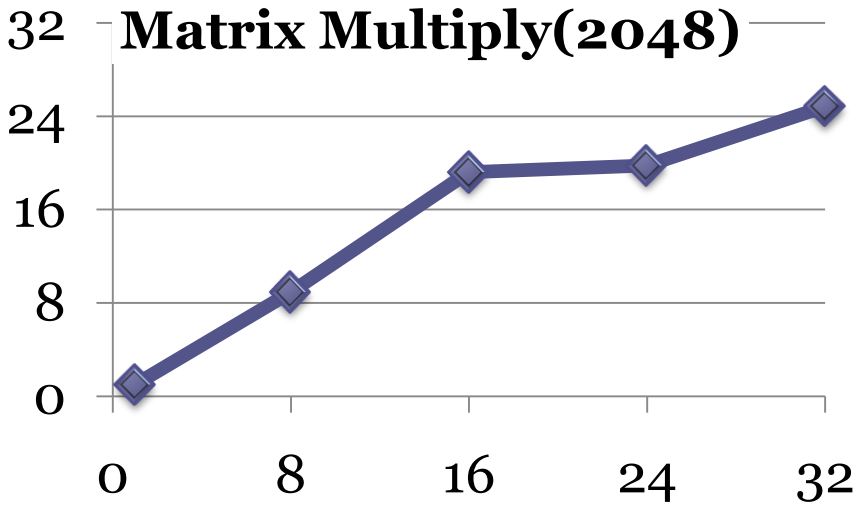
Tasking Layer Implementation

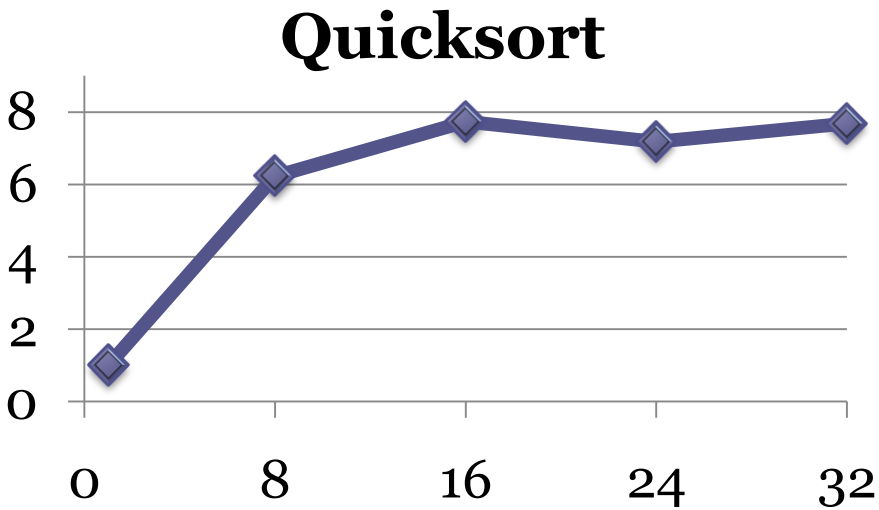
- Written as a simple wrapper of MassiveThreads
- Support multiple locales without extra coding
  - Thanks to pthread-compatible API
  - Internal communication threads can be managed

Single-Locale Performance









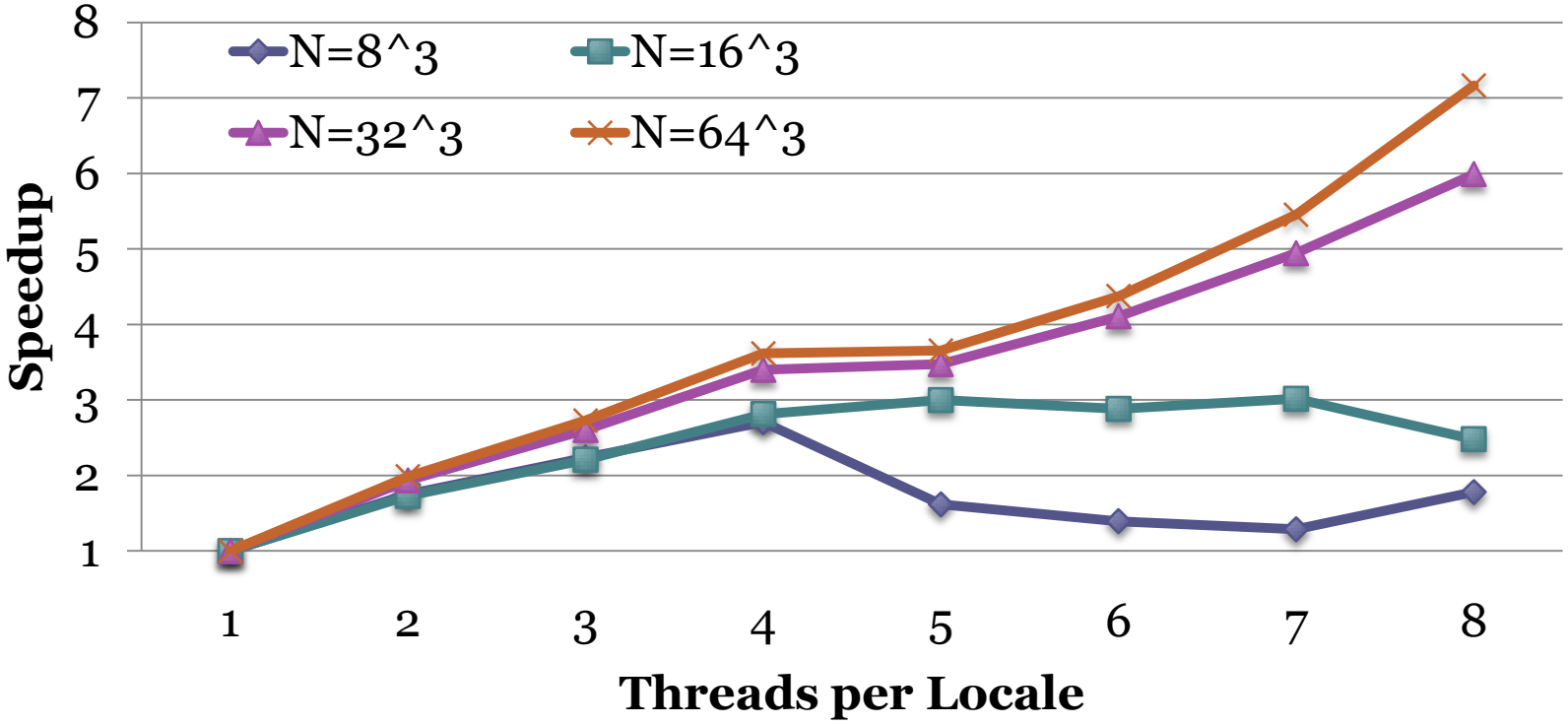
Summary

- Chapel tasking layer by MassiveThreads :
  - Good recursive task-parallel performance
  - Support multiple locales easily
- Future work:
  - More performance studies
  - Integrating the library into forall
  - Support multiplexing for more I/O types
- MassiveThreads source code is available!
  - http://code.google.com/p/massivethreads/
  - License: initially LGPL, change to new BSD after review

Other Ongoing Efforts : Molecular Dynamics in Chapel

- Programmability
  - Parallelization
    - Less than 5% code modification from serial version
  - GPGPU integration
- Performance implication
- Feedback for both users and developers
- Open source
  - http://mdoch.googlecode.com/

Other Ongoing Efforts : Molecular Dynamics in Chapel



http://mdoch.googlecode.com/

Thank you for your attention!