Motivation

• HPCG is becoming increasingly important in the HPC world

• Few optimized versions are open-source
  • None are Arm-specific

• Establish a baseline codebase for the community to build upon

• Stop reinventing the wheel
Inspiration

• We have used different sources as inspiration
Multi-level task dependency graph

- Nodes in the same level of the graph can be processed in parallel
- How to:
  1. Add node 0 to the level 1
  2. Mark node 1 as visited
  3. Close level 1
  4. Check neighbors of nodes in previous level to see if dependencies are fulfilled
     1. If yes, add node to the level and mark node as visited
     2. If no, continue with the next node
  5. Close level, add new level and go to 4 if no more nodes to process

Level 0: 0
Level 1: 1
Level 2: 2, 4
Level 3: 3, 5
Level 4: 6, 8
Level 5: 7, 9
Level 7: 10, 12
Level 8: 11, 13
Level 9: 14
Level 10: 15
Block multi-coloring

- Blocks with the same color can be processed in parallel
- How to:
  1. Group N consecutive nodes in blocks
  2. Colorize blocks
  3. Reorder blocks and rows of blocks sharing the same color
Merging all together

Finest level

Coarser levels
• Experiments executed on a dual-socket state-of-the-art Arm platform
Multi-node performance

- **Optimized:**
  - 8 MPI ranks per node
  - 7 OpenMP threads per MPI rank
  - 256x224x256

- **Vanilla:**
  - 56 MPI ranks per node
  - OpenMP disabled
  - 128x128x128
Summary

• The code is open-source
  • https://gitlab.com/arm-hpc/benchmarks/hpcg
  • We welcome contributions! 😊

• We can collaboratively improve things
  • Hand-made NEON code
  • Hand-made SVE code
  • Platform-specific optimizations
  • Network communication

• Further information about our code on the the Arm Community blog
  • https://community.arm.com/tools/hpc/b/hpc/posts/parallelizing-hpcg
Thank You
Danke
Merci
謝謝
ありがとう
Gracias
Kiitos
감사합니다
धन्यवाद
תודה