Task Mapping for Non-contiguous Allocations

David P. Bunde¹, Johnathan Ebbers¹, Stefan P. Feer², Vitus J. Leung³, Nickolas W. Price¹, Zachary D. Rhodes⁴, Matthew Swank¹

¹Knox College 
[d bunde,j ebbers, np rice, m swank]@knox.edu
²3M Health Information Systems 
sfeer@m m.com
³Sandia National Laboratories 
vj leung@sandia.gov
⁴Allstate Corporation 
rhodes87@gmail.com

Task Mapping

- Assign job tasks to allocated nodes
- Close communicating tasks use fewer links
- Preserves bandwidth, lessening interference between jobs
- Applications have seen a 1.64x speedup from improved mapping [4]
- Potentially saves power

We map common job communication pattern (stencil) onto practical allocations for Cray X (non-contiguous mesh nodes)

Algorithms

Rotations
- Rotate job so relative order of dimension lengths matches dimensions of bounding box of nodes
- Applied to all algorithms except Baseline and Grouping

Baseline (provided by ALPS and Moab)
- Number cores in alloc. order and tasks in row-major order
- Map corresponding elements

Grouping [3]
- Baseline after grouping tasks into 2x2x4 blocks

ColMajor
- Number tasks and cores in column-major order
- Map corresponding elements

RowMajor
- As ColMajor, but using row-major order
- Map corresponding elements

Ordered
- Try different linear orders (inc. ColMajor and RowMajor)
- Take whichever gives lowest average hops

Corner (generalizes “Expands from corner” [3] w/ rotations)
- Number tasks and cores based on distance from (0,0)
- Map corresponding elements

Corner (generalizes “Corner to center” [3] w/ rotations)
- Similar to Corner, but switches between corners

Overlay
- Identifies desired location for each task
- Lower left corner task desires lower left corner core
- Other tasks relative to this
- Each task is mapped as near as possible to desired location

TwoWayOverlay
- Similar to Overlay, but switches between opposite corners

RCB
- Bisects job’s longest dimension and bisects cores w/ same one
- Recursively map elements of each half
- [3] also has bias heuristic, but with general graphs
- Loses geometric relationship between recursion levels so doesn’t perform as well in our setting

References


Performances

Best overall performance: RCB with 2 cores/rank (best in 17 of 20 trials at 8K+ cores (5 runs of 4 sizes))

Rotations and correlations to enable future simulations
- Rotation step improves ~60% of runs (ave. benefit 1-6%)
- Average hops between communicating tasks correlates with running time (Spearman rank test; significance level 0.05 or better for 14x4 cores)
- Small jobs affected by hop between sockets at node
- Allowed trace-based simulations that confirmed relative quality of mapping algorithms

Experiments

Machine: Cielo – Cray XE6
- 8,944 compute nodes (2 sockets/node, 8 cores/socket)
- Gemini 3D torus in 1x1x12x4 (XYZ) topology
- 6.74x3.85x4.38 (XYZ) Torus bisection bandwidth
- #22 on June 2013 Top500 list

Job configurations
- 16-64 cores, with rectangular prism job shape
- Between 1 and 16 cores per MPI rank (MPI + OpenMP)

Application: miniGhost “mini application” [2]
- Boundary exchange using stencil computation
- Modeled after CTH, a multi-material, large deformation, strong shock wave, solid mechanics code
- Gives similar scaling as CTH, including relative performance improvement for remapping

Results averaged over 5 runs

Overall results
- RCB is consistently the best mapper
- 4 cores/rank: 16% ave. improvement at 64K (12.8-14.6% per run)
- Better w/ other cores/ranks: 24.1% at 2 and 28.4% at 1
- Also most consistent: Baseline running time had multi-second std. deviations while RCB always 0.6-0.9 seconds

- Best overall performance: RCB with 2 cores/rank

Time as a function of job size (4 cores/rank)

Time for 64K-core job as a function of cores/rank

Number of cores per MPI rank
- Tuning option in miniGhost (changes number of ranks)
- More ranks means more messages, but smaller
- More ranks also slows collective operations
- Worst mappers favor 16 cores/rank
- Fewer messages helps network-bound runs
- Better mappers perform best at intermediate values
- We focus on 4 cores/rank; also favored in previous work [1]