JCAHPC

Oakforest-PACS
チューニングガイド

最先端共同HPC基盤施設 (JCAHPC)

Intel Parallel Studio 2017 update 1 用

2017/1/27

東京大学情報基盤センター
INFORMATION TECHNOLOGY CENTER, THE UNIVERSITY OF TOKYO

筑波大学
計算科学研究センター
Center for Computational Sciences

コンパイル

必須

- -xMIC-AVX512 -qopenmp

(フラットMPIなら-qopenmpは無い方が速い)

やった方がいい

- -align array64byte
  - キャッシュラインサイズにalign

試した方がいい

- -qopt-streaming-stores= always / never / auto
  - デフォルトはauto
  - Stream benchではalwaysを指定
  - 文ごとにも制御可能
  - #pragma vector nontemporal (in C/C++)
  - !DEC$ vector nontemporal (in F)

実行時オプション（共通）

補助コマンド

- numactl
  - affinity設定用
  - MCDRAM指定にも使う
  - numactl -s $$: 使用コア
  - numactl -H : HW情報
- taskset
  - 使用コア制限に便利
  - taskset -c 2-64,67-127,130-191,194-255 ./a.out: 使用するコアを明示

環境変数

export コマンドで設定（以下では省略）

- I_MPI_XXX
  - Intel MPI固有のオプション
  - KMP_XXXより優先される
- KMP_XXX
  - Intel Compiler固有のオプション
- OMP_XXX
  - OpenMPランタイムオプション（汎用）

実行時オプション（affinity 1/2）

デバッグ中は必須

思ったように割り当てできているか確認

- I_MPI_DEBUG=5
  - 各ランクへの計算ノードおよびコアの割り当て状況
- KMP_AFFINITY=verbose
  - 各プロセス内でのコアの割り当て状況
- TMI_DEBUG=1
  - プロトコルにtmiを指定した時のデバッグ情報

Tickless設定

OSジッタを避けるため、コア0（およびそのHTコア）だけがタイマ割り込みを受け取る設定になっている

- KNLのコア0,1（タイル1単位）は使用を避ける
- I_MPI_PIN_PROCESSOR_EXCLUDE_LIST=0,1,68,69,136,137,204,205
- MPI未使用のとき:
  KMP_HW_SUBSET=64c@2,1t
  - 最初の2コアを使わずに64コア確保

実行時オプション (affinity 2/2)

物理コアに1つずつ割り当て (HyperThreadingコアは使わない想定)

- KMP_HW_SUBSET=1T
- unset KMP_AFFINITY (すでにcompactが指定されているのでunsetを忘れると性能が悪化する)

物理コアに2つ(以上)ずつ割り当て(HyperThreadingコアを使う)

- KMP_HW_SUBSET=2T
  - 4Tまで可能
- KMP_AFFINITY=compact を合わせて指定する方がよい(すでにされている)

実行時オプション (ノード当たり複数MPIプロセス) (1/2)

PPN (Process Per Node) が2以上の場合

- ノード当たりに複数(N)割り当て
  - I_MPI_PERHOST=N または
  - mpiexec.hydra, mpirunの引数に -ppn N
- プロセス単位での割り当て方
  - I_MPI_PIN_DOMAIN=PN
  - 例えば全体で256コア使いたければ PN=256/N
  - 例は次ページ参照

実行時オプション (ノード当たり複数MPIプロセス) (2/2)

例:
PPN=4 (4MPI 16スレッド)

- OMP_NUM_THREADS=16
- I_MPI_PIN_DOMAIN=64
- I_MPI_PERHOST=4

PPN=8 (8MPI 8スレッド)のとき

- OMP_NUM_THREADS=8
- I_MPI_PIN_DOMAIN=32
- I_MPI_PERHOST=8

実行時オプション（MCDRAM関連）

- numactl --membind=1 ./a.out
  - MCDRAMに入りきらない場合は numactl --preferred=1 ./a.out とするとベストエフォートでMCDRAMを使う
  - numactl --interleave=0,1 ./a.out とするとMCDRAMとDDR4をround-robinで使う

MPIバッファなどをMCDRAM上に

（DDR4よりややレイテンシ大なので注意）

- I_MPI_HBW_POLICY=<A>,<B>,<C>
  - 選択肢: hbw_bind / hbw_preferred / hbw_interleave / 書かない
  - <A>: ユーザプロセス
    - これがあれば左のnumactlは不要
  - <B>: MPIバッファ
  - <C>: Win allocate (MPI-3)

実行時オプション（通信関連）

プロトコルスタック選択

- I_MPI_FABRICS_LIST=tmi
  - tmi: OmniPathの推奨
  - ofi: 今後の業界標準, tmiよりいい場合があるかもしれない
  - ofa: 動くが遅い、デバッグ用（IB互換モード）
- I_MPI_FABRICS=shm:tmi
  - shm: 共有メモリ（ノード内）
  - shmの代わりにtmiにした方が良い場合も
    - つまり I_MPI_FABRICS=tmi:tmi

試した方がいい

- HFI_NO_CPUAFFINITY=1
  - 差が出る場合と出ない場合がある？
    - ノード内複数プロセス実行時などは指定した方がいいかも
  - tmiを選んだ場合に影響（?? 未確認）

確認：各実行モードのnumactl -H

```
available: 2 nodes (0-1)
node 0 cpus: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43
44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87
88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122
123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153
154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184
185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215
216 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246
247 248 249 250 251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271
node 0 size: 98147 MB
node 0 free: 85936 MB
node 1 cpus:
node 1 size: 16384 MB
node 1 free: 15799 MB
node distances:
node   0   1
  0:  10  31
  1:  31  10
```

FLAT QUADRANT

- CPUはnode 0のみ
- DDR4はnode 0
- MCDRAMはnode 1

available: 1 nodes (0)
node 0 cpus: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57
58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86
87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111
112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132
133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153
154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174
175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195
196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216
217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237
238 239 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 256 257 258
259 260 261 262 263 264 265 266 267 268 269 270 271
node 0 size: 98147 MB
node 0 free: 85492 MB
node distances:
node 0
0: 10

CACHE QUADRANT

- CPUもメモリも1つだけ