Sandia National Laboratories

Exceptional service in the national interest

CGNS PARALLEL DECOMPOSITION

A WORKFLOW FOR USING CGNS IN PARALLEL HPC ANALYSES

Gregory Sjaardema, Simulation Modelling Sciences

HDF5 User Group (HUG) 2023

August 16-18, 2023
Scott Laboratory, Columbus, OH

Unclassified Unlimited Release

A supercomputer is a device for turning compute-bound problems into I/O-bound problems

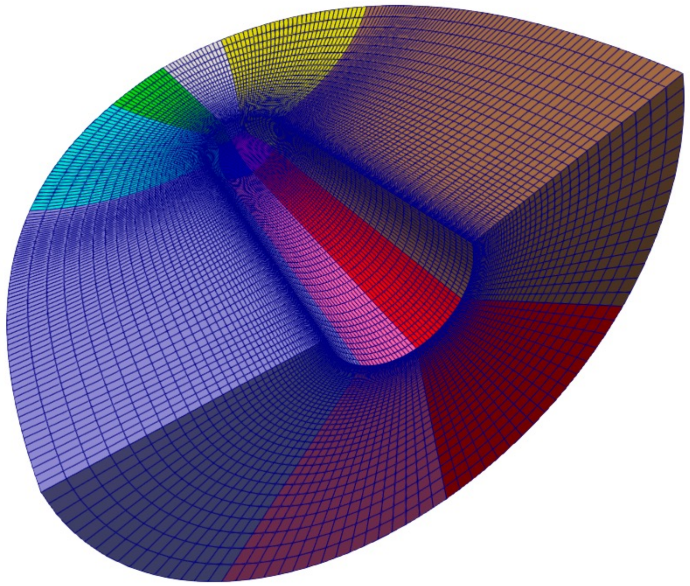
Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia LLC, a wholly owned subsidiary of Honeywell International Inc. for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

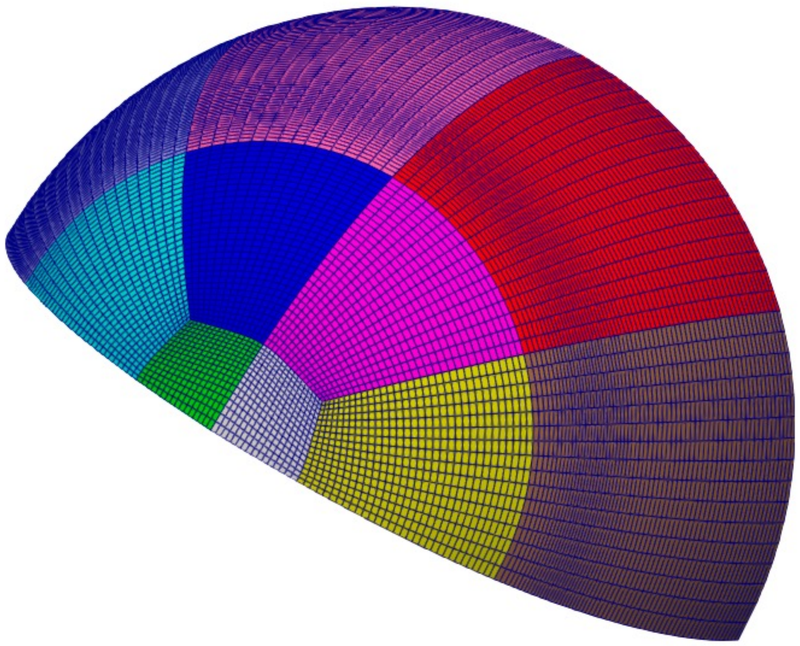
Request ID **1717606**

U.S. DEPARTMENT OF ENERGY

NNSA

CGNS PARALLEL ANALYSIS OPTIONS





Unstructured CGNS

- Auto-decomp (1 file run on N ranks)
  - Decomposition strategy / algorithm
  - Zone is empty or contiguous on a processor rank

- File-per-rank (N files run on N ranks)
  - Reconstruction algorithm –
    - order based on inter-zone grid connectivity
    - Based on left/right/upper/lower/front/back can order all zones
    - Based on ordering, can get global size of a zone
    - Can set offset of a local zone into the global zone

Structured CGNS

- Zoltan decomposition options, similar to Exodus

RELATIONSHIP TO HDF5

CGNS uses HDF5

THG supports development of CGNS

Once model is decomposed, HDF5 parallel read capabilities are used

- Each rank reads its portion of the mesh
- All ranks are reading at same time the information that they need.

REQUIREMENTS

Distribute Work Evenly

No communication during decomposition

Decomposition running time independent of cell count.

Minimize inter-processor communication

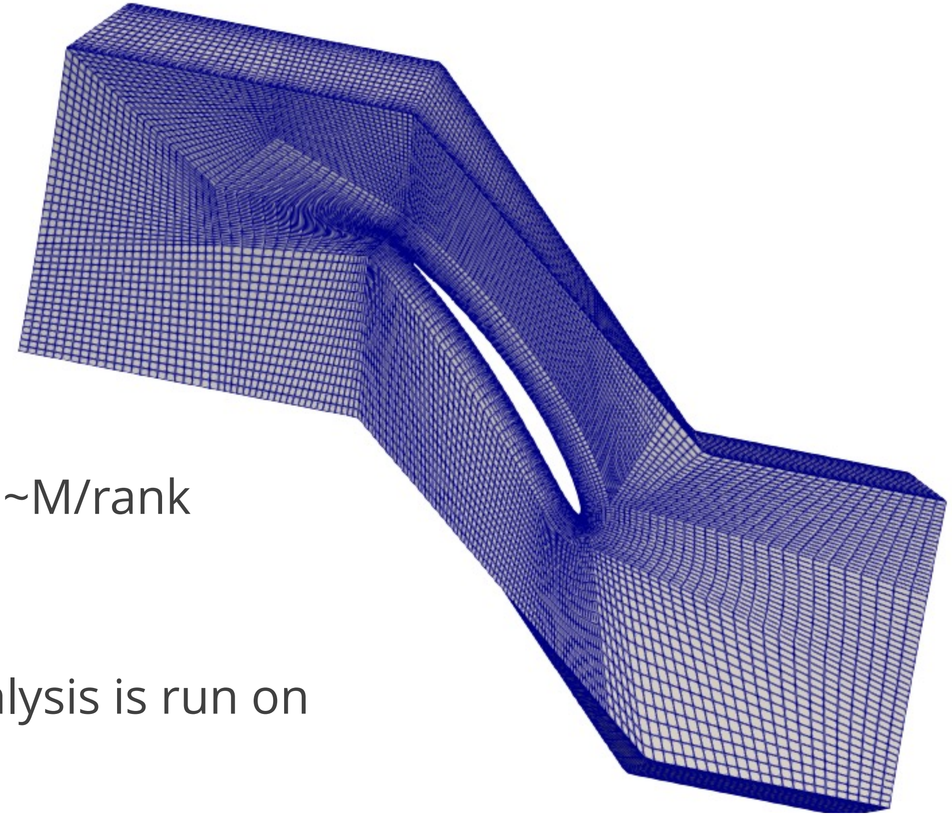
Memory efficient

- cannot hold entire mesh on single rank
- If a serial run can handle M cells,
- then an N-rank run should be able to handle ~M*N cells or ~M/rank

Consistent / Invisible

- The number of ranks does not affect model metadata.
- The user should not need to know how many ranks the analysis is run on
- Intelligent default behavior

Visible if needed – if there are problems, make it easy to determine what went wrong

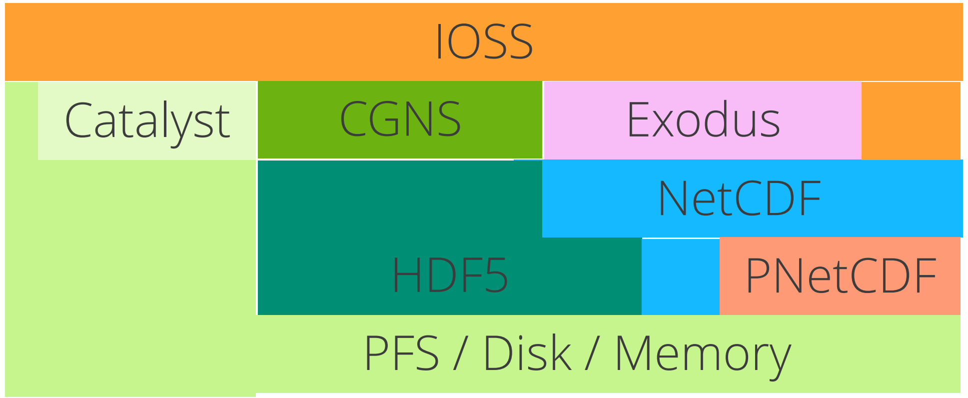


– I/O SUBSYSTEM – IOSS LIBRARY

- Started as the IO component of the Sierra project – 12/1999
- Provide a database-independent interface to Sierra shielding the applications from differences in database types (Exodus, CGNS, XDMF, Adios2, Catalyst, …)
- Supports Advanced HPC Capabilities:
  - Kokkos Data
  - Burst Buffer
  - Data Warehouse (FAODEL)
  - Embedded Visualization (Catalyst2)

- Auto-decomposition option replaces the legacy file-per-processor mode
  - Uses either HDF5 or PnetCDF for parallel input
  - Uses decomposition methods in Zoltan and ParMETIS
  - Supports Exodus and CGNS (Structured and Unstructured)

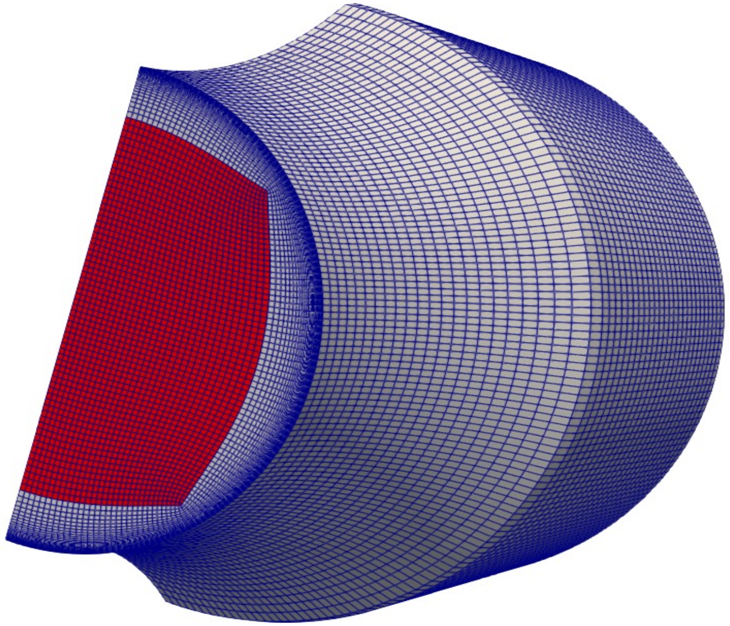
- Auto-join (single file output) option
  - Uses HDF5 or PnetCDF for parallel output
  - Scalability issues…. Being addressed.



DECOMPOSITION ALGORITHM

Input:

- Number of ranks
- Load Balance Factor (LBF): ”goodness” of decomposition



Calculate:

- Average_work = #cells / #ranks
- Decompose such that:
  - `Average_work / LBF < work/rank < Average_work * LBF`

Overview:

- Pre-split -- Give the algorithm at least #rank zones of approximately correct size
- Adapt pre-split if needed
- Assign zones to ranks

PRE-SPLIT: NEED #ZONES >= #RANKS

Pre-split:

- Per-zone: `splits[zone] = work[zone] / average_work`
- `If (sum_splits != #rank)` adjust_splits
  - Pick zone 'zone' with Minimum `abs(average_work – work[zone] / (splits[zone] +/- 1))`
  - Repeat until "`sum_splits == #rank`"
- Check that this set of splits gives work for each zone close to average

Split the zones to get sub-zones of about right size…

- if `splits[zone] == power-of-2` (2^n)
  - Split in half `n` times
- Else
  - Split off such that remainder can be split in the power-of-2 mode…

ZONE SPLITTING

Split StructuredZone along the largest ordinal into two children; return the created zones.

- Input: avg_work – we want one of the children to have close to that much work.
- *Split Ratio* = avg_work / work
- Which ordinal gives work closest to avg_work
- Try to keep as “squarish” as possible
- Try to keep >1 interval on each ordinal

Split zones know:

- Adam, parent, sibling
- Size and offset into adam zone
- What ordinal they were split from parent

Add a ZGC due to splitting – siblings communicate

Add ZGC from parent

Ranks: 4 (avg=16)

- Adam (8x8x1)
  - P1(8x4x1)
    - P1C1 (4x4x1)
    - P1C2 (4x4x1)
  - P2 (8x4x1)
    - P2C1 (4x4x1)
    - P2C2 (4x4x1)

Ranks: 3 (avg = 21.3)

- Adam (8x8x1)
  - P1 (8x3x1)
  - P2(8x5x1)
    - P2C1 (4x5x1)
    - P2C2 (4x5x1)

```
Split Zone block_1 (1) Adam block_1 (1) with intervals       8 8 1,     work =           64, offset 0 0 0, ordinal 0, ratio 0.667
        Child 1: Zone block_1_c1 (2) with intervals       5 8 1,       work =           40, offset 0 0 0
        Child 2: Zone block_1_c2 (3) with intervals       3 8 1,       work =           24, offset 5 0 0

Split Zone block_1_c1 (2) Adam block_1 (1) with intervals       5 8 1, work =           40, offset 0 0 0, ordinal 1, ratio 0.500
        Child 1: Zone block_1_c1_c1 (4) with intervals       5 4 1,     work =           20, offset 0 0 0
        Child 2: Zone block_1_c1_c2 (5) with intervals       5 4 1,     work =           20, offset 0 4 0
```

Work: 64
#ranks: 4
Average_Work = 64/4 = 16

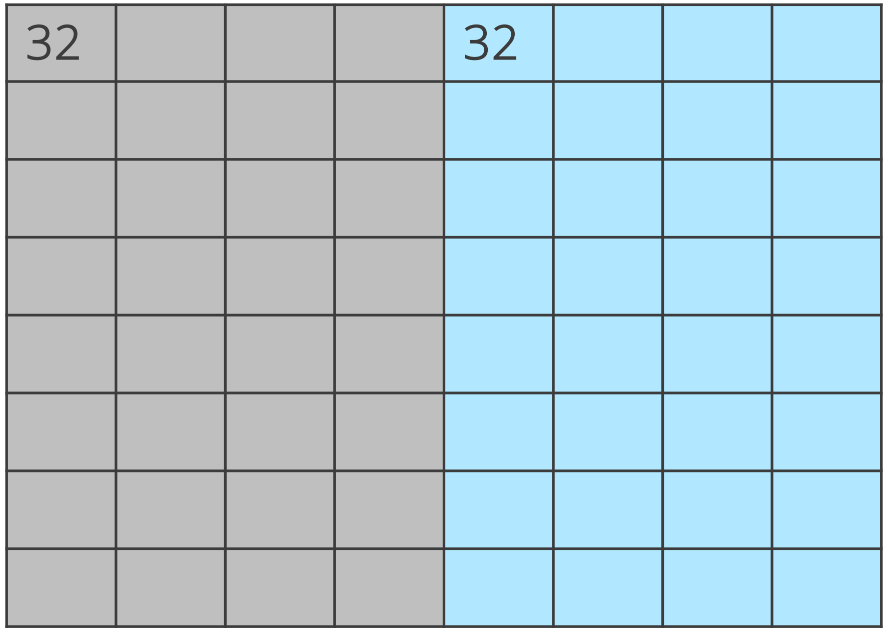
64							

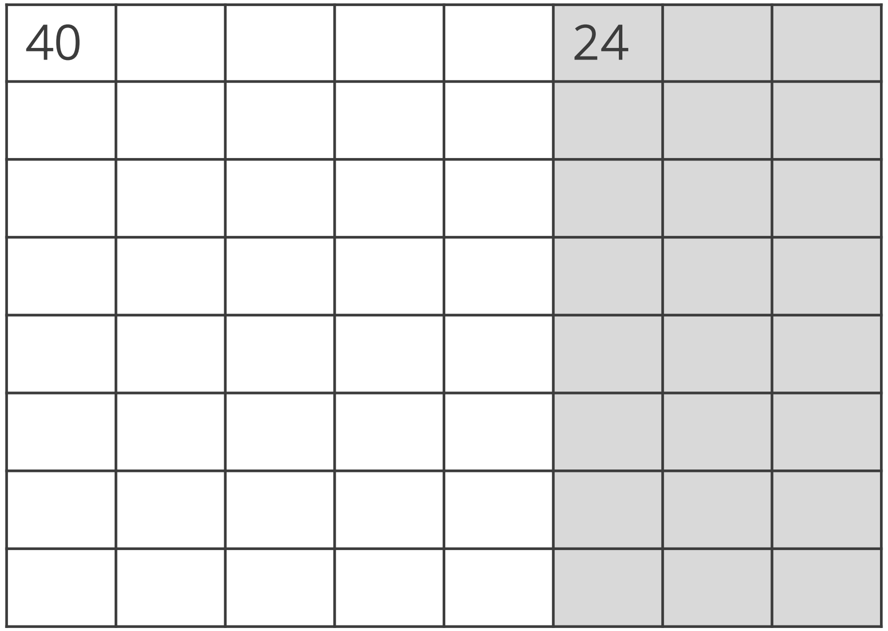
Work: 64
#ranks: 3
Average_Work = 64/4 = 21

64							

Pre-Split:
Splits = 64 / 16 = 4 (2^2)
Split 1,2: split in half
Split 3,4: split each half in half

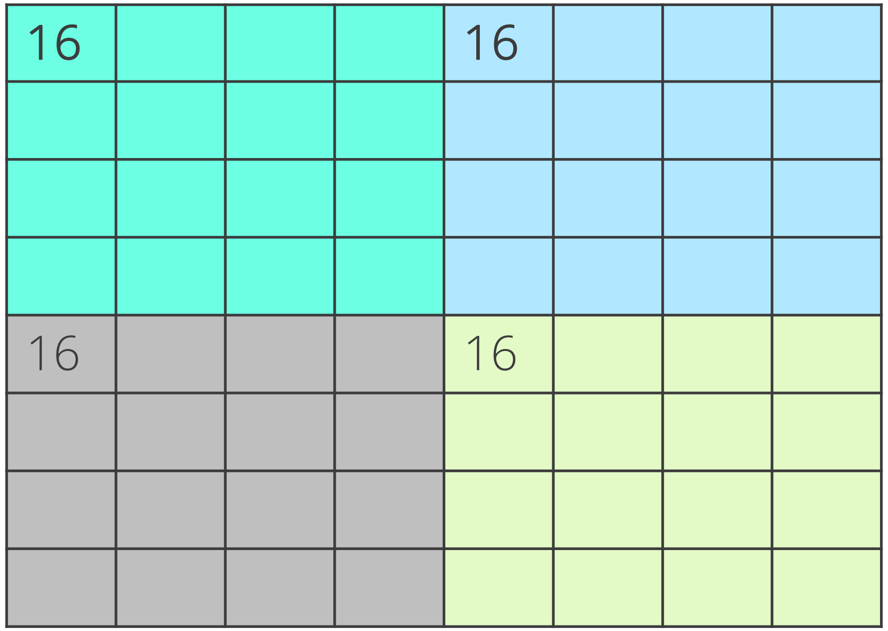
Pre-Split:
Splits = 64 / 21 = 3
Split 1,2: 8 * 1/3 = 2.7 ~ 3.
Split 1: 3x8 = 24, Split 2: 5x8 = 40

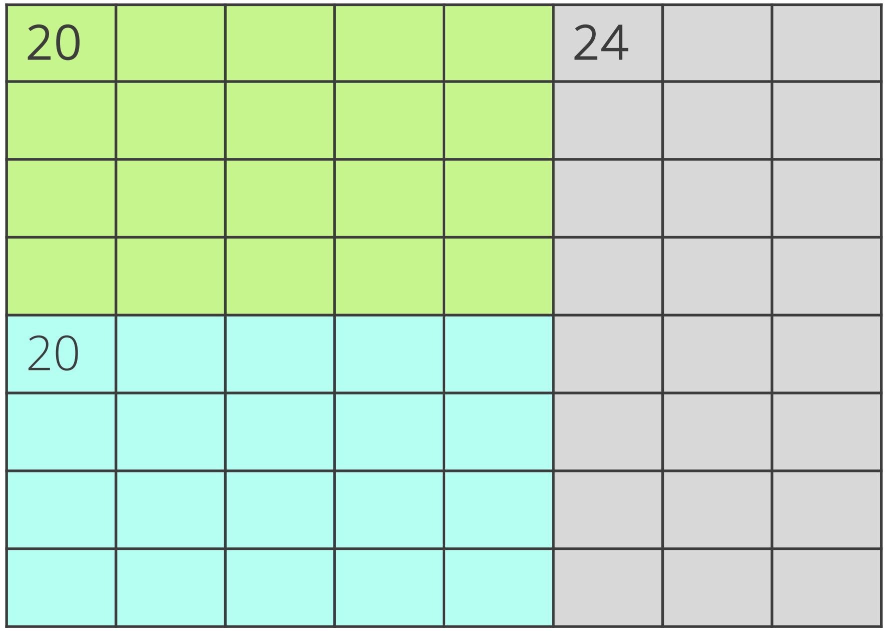




Pre-Split:
Splits = 64 / 16 = 4 (2^2)
Split 1,2: split in half
Split 3,4: split each half in half
Each split has work=16

Pre-Split:
Splits = 64 / 21 = 3
Split 1,2: 8 * 1/3 = 2.7 ~ 3.
Split 1: 8x3 = 24, Split 2: 8x5 = 40
Split 2: 40 / 21 ~ 2
Split 2, 3: 4x5
Work: 24, 20, 20





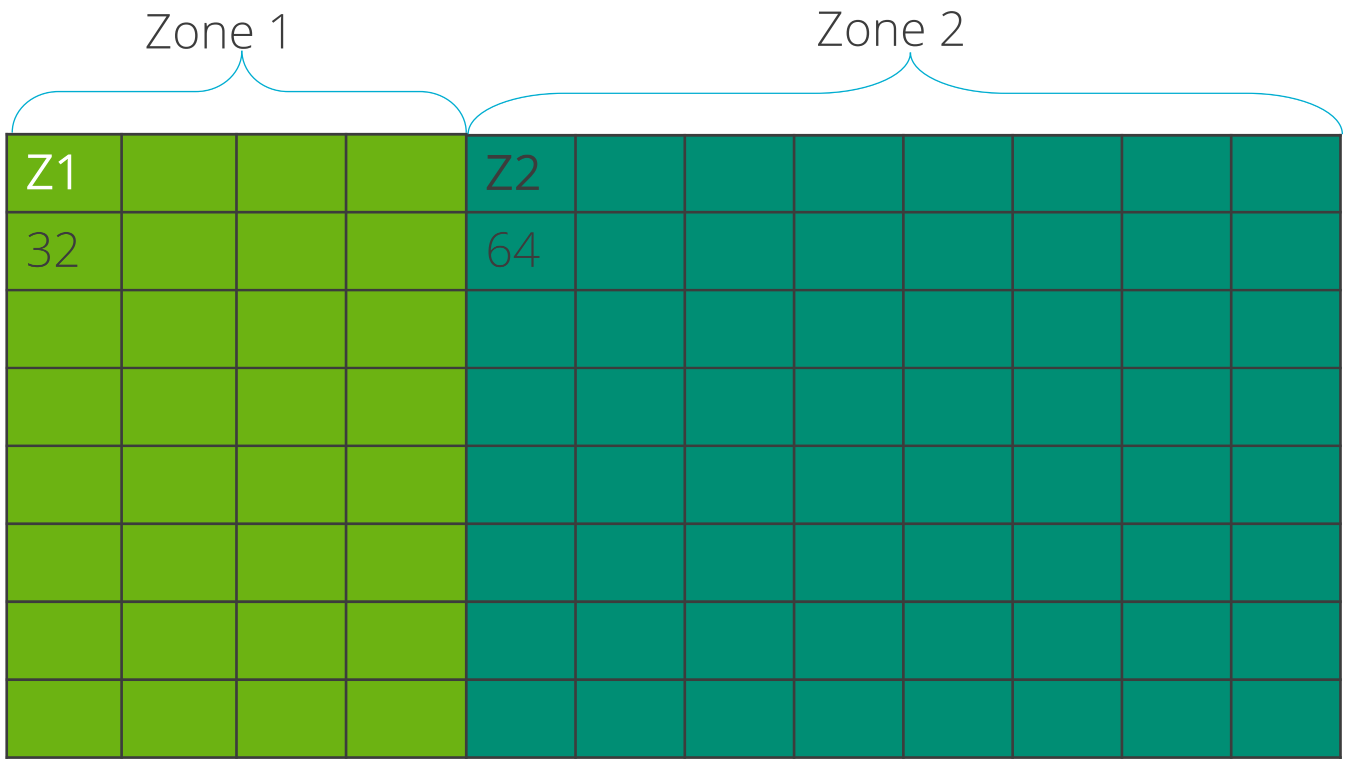
ASSIGN ZONES TO PROCS:

Until `zone->work() > avg_work * load_balance_threshold`

- Sort zones based on work.
- Assign first #proc zones to procs 1..#procs
- Continue with remaining zones:
  - Find proc with minimum work and zone(s) on it do not match this zones Adam
- Split zone(s) if cannot get a proc's work within range
  - Want work[proc] / avg_work <= LBF
- Count #procs where work exceeds desired and split that many zones;
  - Split largest zone on a proc which exceeds desired work range
- Repeat until no rank exceeds desired

Execution time dominated by max work, so better to have 1 rank much below average than 1 rank much more than average.

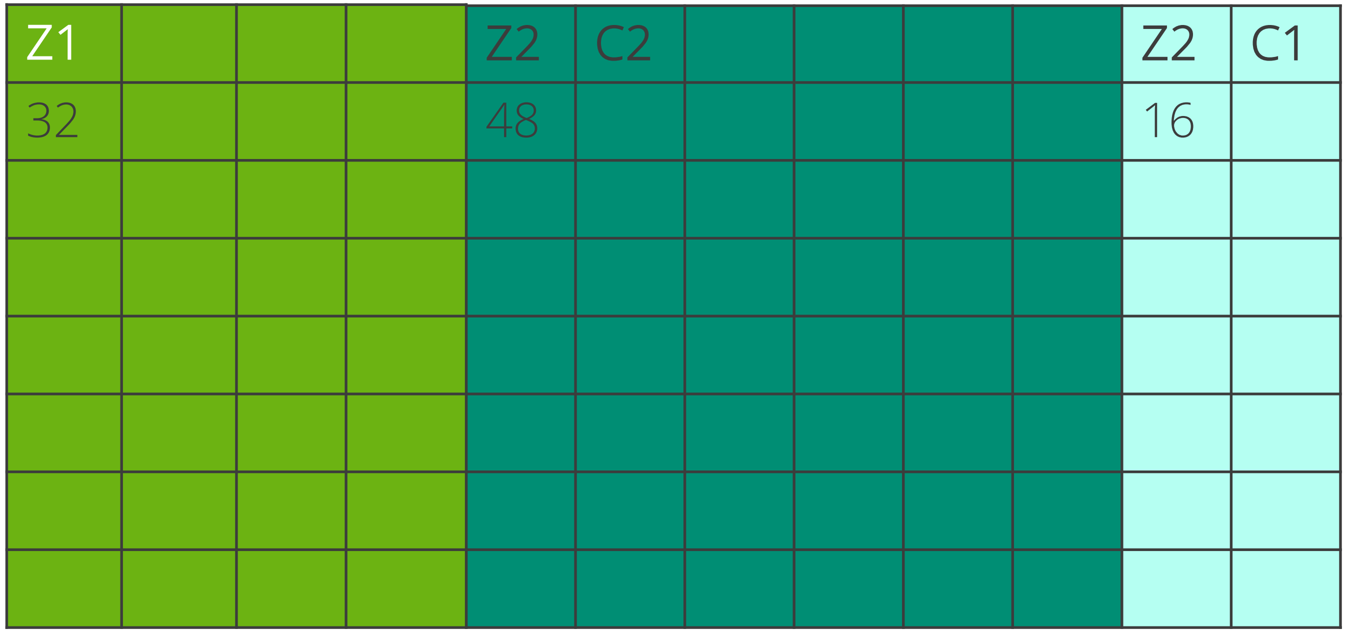
- Z1 work = 32
- Z2 work = 64
- Work = 96
- #ranks = 4
- Average_Work = (32 + 64) / 4 = 24



Work: 96 (zone 1: 32, zone 2: 64)
#ranks: 4
Average_Work = 96/4 = 24

Pre-Split:

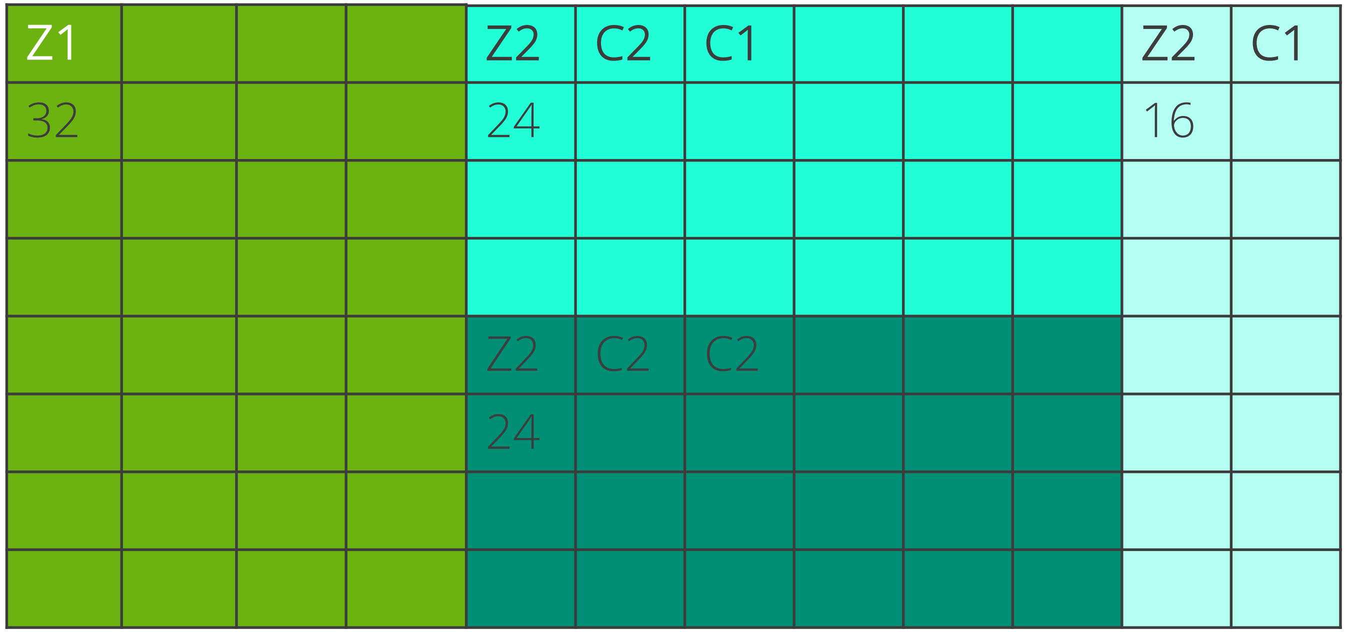
- Zone 1: 32 / 24 = 1.3 = ~1
- Zone 2: 64 / 24 = 2.7 = ~3
  - Split 1: 8x2 (16)
  - Split 2: 8x6 (48)



Work: 96 (zone 1: 32, zone 2: 64)
#ranks: 4
Average_Work = 96/4 = 24

Pre-Split:

- Zone 1: 32 / 24 = 1.3 = ~1
- Zone 2: 64 / 24 = 2.7 = ~3
    - Split 1: 8x2 (16)
    - Split 2,3: 4x6 (24)



Work: 96 (zone 1: 32, zone 2: 64)
#ranks: 4
Average_Work = 96/4 = 24

Zone 1 is too large: 32 /24 = 1.33

- Split Z1, C1: 6x4 = 24
- Split Z1, C2: 2x4 = 8

Pre-Split:

- Zone 1: 32 / 24 = 1.3 = ~1
- Zone 2: 64 / 24 = 2.7 = ~3
    - Split 1: 8x2 (16)
    - Split 2,3: 4x6 (24)

Work: 96 (zone 1: 32, zone 2: 64)
#ranks: 4
Average_Work = 96/4 = 24

Pre-Split:
- Zone 1: 32 / 24 = 1.3 = ~1
- Zone 2: 64 / 24 = 2.7 = ~3
    - Split 1: 8x2 (16)
    - Split 2,3: 4x6 (24)

Zone 1 is too large: 32 /24 = 1.33
- Split Z1, C1: 6x4 = 24
- Split Z1, C2: 2x4 = 8

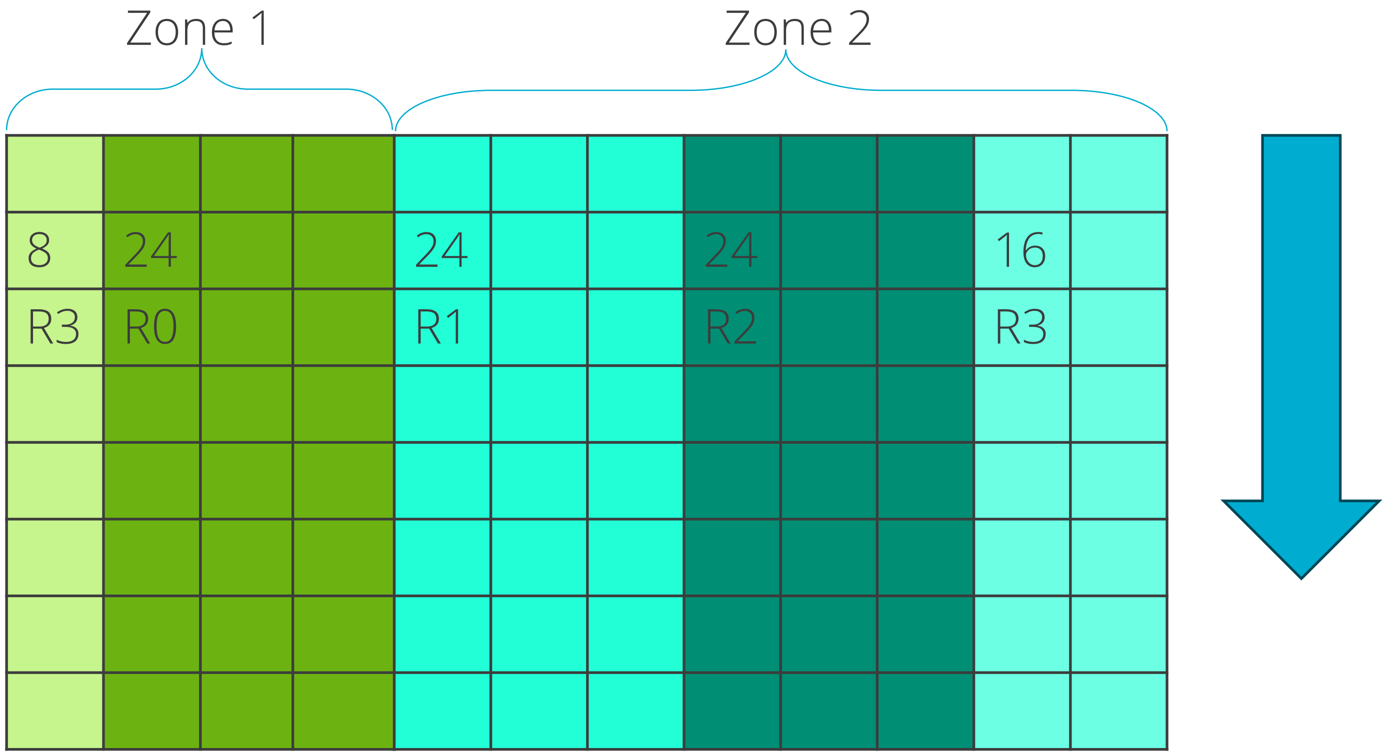
Assign Zones to Ranks:
- R0: Z1C1 – 24
- R1: Z2C2C1 – 24
- R2: Z2C2C2 – 24
- R3: Z2C1 – 16, Z1C2 – 8 (24)

SPECIAL OPTIONS – LINE DECOMP

Can specify 1 or 2 ordinals which a zone should not be split

- This is so a “line solver” can keep all cells in a column from the specified surface on the same rank. Speeds up solver since no communication on column of cells.

Minimum ordinal size – currently set to 2 but will override if cannot generate a good decomposition. Enhancement is to make user-settable.



CGNS_DECOMP

Would like to know "goodness" of a decomposition prior to running analysis code

Gives statistics about a parallel decomposition

Runs in serial, but uses same algorithm as used in parallel

Statistics:

- What processor(s) is a zone assigned to
  - Size on the processor
  - "Surface expansion"
- Histogram showing work per rank
- Work histogram showing mean and median
- Communication map – what ranks communicate with each other for each zone
- Decomposition penalty – max work / avg work

```
Database: multiple_zones_fields-out.cgns
Mesh Type = Structured, CGNS
                                                                   Variables : Transient / Reduction
Spatial dimensions =    3                                          Global     =  0         0
Node blocks        =    1      Nodes          =  1,350             Nodal      =  0         0
Edge blocks        =    0      Edges          =      0             Edge       =  0         0
Face blocks        =    0      Faces          =      0             Face       =  0         0
Element blocks     =    0      Elements       =      0             Element    =  0         0
Structured blocks  =   14      Cells          =  1,024             Structured =  3         0
Node sets          =    0      Node list      =      0             Nodeset    =  0         0
Edge sets          =    0      Edge list      =      0             Edgeset    =  0         0
Face sets          =    0      Face list      =      0             Faceset    =  0         0
Element sets       =    0      Element list   =      0             Elementset =  0         0
Element side sets  =    4      Element sides  =    592             Sideset    =  0
Assemblies         =    0                                          Assembly   =  0         0
Blobs              =    0                                          Blob       =  0         0

Time steps         =    6
```

```
Decomposing 14 zones over 11 processors; Total work = 1,024; Average = 93 (goal)

        Zone: blk-01      Proc:  6      Ord:  8 x 2 x 2      Work:   32 (unsplit)
        Zone: blk-02      Proc:  7      Ord:  8 x 2 x 2      Work:   32 (unsplit)
        Zone: blk-03      Proc:  8      Ord:  8 x 2 x 2      Work:   32 (unsplit)
        Zone: blk-04      Proc:  9      Ord:  8 x 1 x 4      Work:   32 (unsplit)
        Zone: blk-05      Proc:  0      Ord:  8 x 4 x 4      Work:  128 (unsplit)
        Zone: blk-06      Proc:  1      Ord:  8 x 4 x 4      Work:  128 (unsplit)
        Zone: blk-07      Proc:  2      Ord:  8 x 4 x 4      Work:  128 (unsplit)
        Zone: blk-08      Proc: 10      Ord:  8 x 2 x 2      Work:   32 (unsplit)
        Zone: blk-09      Proc:  6      Ord:  8 x 2 x 2      Work:   32 (unsplit)
        Zone: blk-10      Proc:  7      Ord:  8 x 2 x 2      Work:   32 (unsplit)
        Zone: blk-11      Proc:  8      Ord:  8 x 1 x 4      Work:   32 (unsplit)
        Zone: blk-12      Proc:  3      Ord:  8 x 4 x 4      Work:  128 (unsplit)
        Zone: blk-13      Proc:  4      Ord:  8 x 4 x 4      Work:  128 (unsplit)
        Zone: blk-14      Proc:  5      Ord:  8 x 4 x 4      Work:  128 (unsplit)
```

```
Work per processor:
        Minimum = 32, Maximum = 128, Median = 128, Ratio = 4

        Processor  0, work =   128  (1.38)      ****************************************
        Processor  1, work =   128  (1.38)      ****************************************
        Processor  2, work =   128  (1.38)      ****************************************
        Processor  3, work =   128  (1.38)      ****************************************
        Processor  4, work =   128  (1.38)      ****************************************
        Processor  5, work =   128  (1.38)      ****************************************
        Processor  6, work =    64  (0.69)      ********************
        Processor  7, work =    64  (0.69)      ********************
        Processor  8, work =    64  (0.69)      ********************
        Processor  9, work =    32  (0.34)      **********
        Processor 10, work =    32  (0.34)      **********
```

```
Decomposing 14 zones over 11 processors; Total work = 1,024; Average = 93 (goal)

        Zone: blk-01      Proc:  6     Ord:  8 x 2 x 2     Work:    32 (unsplit)
        Zone: blk-02      Proc:  7     Ord:  8 x 2 x 2     Work:    32 (unsplit)
        Zone: blk-03      Proc:  8     Ord:  8 x 2 x 2     Work:    32 (unsplit)
        Zone: blk-04      Proc:  9     Ord:  8 x 1 x 4     Work:    32 (unsplit)
        Zone: blk-05    is decomposed. Ord:  8 x 4 x 4     Work:   128
              blk-05_c1   Proc:  9     Ord:  2 x 4 x 4     Work:    32    SurfExp: 1.06
              blk-05_c2   Proc:  0     Ord:  6 x 4 x 4     Work:    96    SurfExp: 1.02
        Zone: blk-06    is decomposed. Ord:  8 x 4 x 4     Work:   128
              blk-06_c1   Proc: 10     Ord:  2 x 4 x 4     Work:    32    SurfExp: 1.06
              blk-06_c2   Proc:  1     Ord:  6 x 4 x 4     Work:    96    SurfExp: 1.02
        Zone: blk-07    is decomposed. Ord:  8 x 4 x 4     Work:   128
              blk-07_c1   Proc:  6     Ord:  2 x 4 x 4     Work:    32    SurfExp: 1.06
              blk-07_c2   Proc:  2     Ord:  6 x 4 x 4     Work:    96    SurfExp: 1.02
        Zone: blk-08      Proc: 10     Ord:  8 x 2 x 2     Work:    32 (unsplit)
        Zone: blk-09      Proc:  6     Ord:  8 x 2 x 2     Work:    32 (unsplit)
        Zone: blk-10      Proc:  7     Ord:  8 x 2 x 2     Work:    32 (unsplit)
        Zone: blk-11      Proc:  8     Ord:  8 x 1 x 4     Work:    32 (unsplit)
        Zone: blk-12    is decomposed. Ord:  8 x 4 x 4     Work:   128
              blk-12_c1   Proc:  7     Ord:  2 x 4 x 4     Work:    32    SurfExp: 1.06
              blk-12_c2   Proc:  3     Ord:  6 x 4 x 4     Work:    96    SurfExp: 1.02
        Zone: blk-13    is decomposed. Ord:  8 x 4 x 4     Work:   128
              blk-13_c1   Proc:  8     Ord:  2 x 4 x 4     Work:    32    SurfExp: 1.06
              blk-13_c2   Proc:  4     Ord:  6 x 4 x 4     Work:    96    SurfExp: 1.02
        Zone: blk-14    is decomposed. Ord:  8 x 4 x 4     Work:   128
              blk-14_c1   Proc:  9     Ord:  2 x 4 x 4     Work:    32    SurfExp: 1.06
              blk-14_c2   Proc:  5     Ord:  6 x 4 x 4     Work:    96    SurfExp: 1.02
```

```
Work per processor:
        Minimum = 64, Maximum = 96, Median = 96, Ratio = 1.5

        Processor  0, work =    96  (1.03)      ****************************************
        Processor  1, work =    96  (1.03)      ****************************************
        Processor  2, work =    96  (1.03)      ****************************************
        Processor  3, work =    96  (1.03)      ****************************************
        Processor  4, work =    96  (1.03)      ****************************************
        Processor  5, work =    96  (1.03)      ****************************************
        Processor  6, work =    96  (1.03)      ****************************************
        Processor  7, work =    96  (1.03)      ****************************************
        Processor  8, work =    96  (1.03)      ****************************************
        Processor  9, work =    96  (1.03)      ****************************************
        Processor 10, work =    64  (0.69)      **************************
```

Load Balance Factor = 1.4

```
Nodal Inflation:
        Original Node Count = 2,016, Decomposed Node Count = 2,016, Created = 0, Ratio = 1.00

Imbalance Penalty:
        Maximum Work = 128, Average Work = 93, Penalty (max/avg) = 1.38
```

Load Balance Factor = 1.1

```
Nodal Inflation:
        Original Node Count = 2,016, Decomposed Node Count = 2,166, Created = 150, Ratio = 1.07

Imbalance Penalty:
        Maximum Work = 96, Average Work = 93, Penalty (max/avg) = 1.03
```

ADDITIONAL WORFLOW APPLICATIONS

CPUP: Serial application to join file-per-rank structured CGNS into single file

Can add "processor_id" cell variable to file to help visualize decomposition

STRUC_TO_UNSTRUC: Convert a structured mesh into an unstructured mesh

CONCLUSIONS / FUTURE WORK

Implemented an algorithm for efficiently decomposing structured CGNS models

Used in production CFD code for a few years

Used on wide range of model sizes and processor counts

- Billions of cells and 10’s to 100’s thousand ranks…

Future Work:

- Better affinity to minimize communication
- Assign minimum work to zone 0 since it usually has more “busy work”
- See what might be needed for GPUs
- Better elimination of “outlier” rank(s)
- Better control of minimum zone size