AVMesh Format for Block Structured Cartesian Grids

Andy Wissink
4/21/2010

The format defines a hierarchy of nested block levels (Fig 1), where each "block" defines a logically-rectangular region in a common index space.

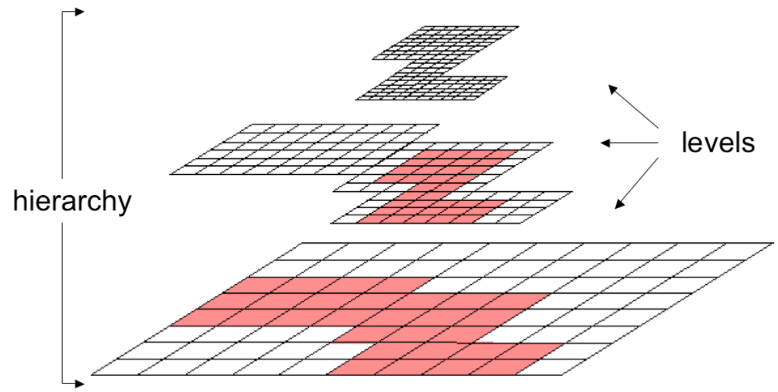

Fig. 1 – Nested hierarchy of block levels used in the block structured format

Although patches will generally not cover an entire level, the index space does (Fig. 2). Because it is a nested Cartesian grid, the only information required to generate the entire grid is the domain and extents of the coarsest level, and the lower/upper indices and level number of each block.

The file format contains the following information:

HEADER:

```
NLEVELS – total number of hierarchy levels
NBLOCKS - total number of blocks
NFRINGES – number of fringe cells
NXC,NYC,NZC – dimensions of coarsest level (level 1)
DOMXLO,DOMYLO,DOMZLO – lower domain extent
DOMXHI,DOMYHI,DOMZHI – upper domain extent
```

Once the number of levels and blocks are known, the block data may be read in:

DATA (BLOCK):

```
IRATIO(NLEVELS) – refine ratio (to coarsest level)
ILO(NBLOCKS,1:3) – lower indices of block
IHI(NBLOCKS,1:3) – upper indices of block
LEVNO(NBLOCKS) – block level number
IBLFLAG(NBLOCKS) – whether block is blanked (0),
     unblanked (1), or contains a mix (2)
BDRYFLAG(NBLOCKS) – whether block resides on
     interior (0) or outer boundary (1)
```

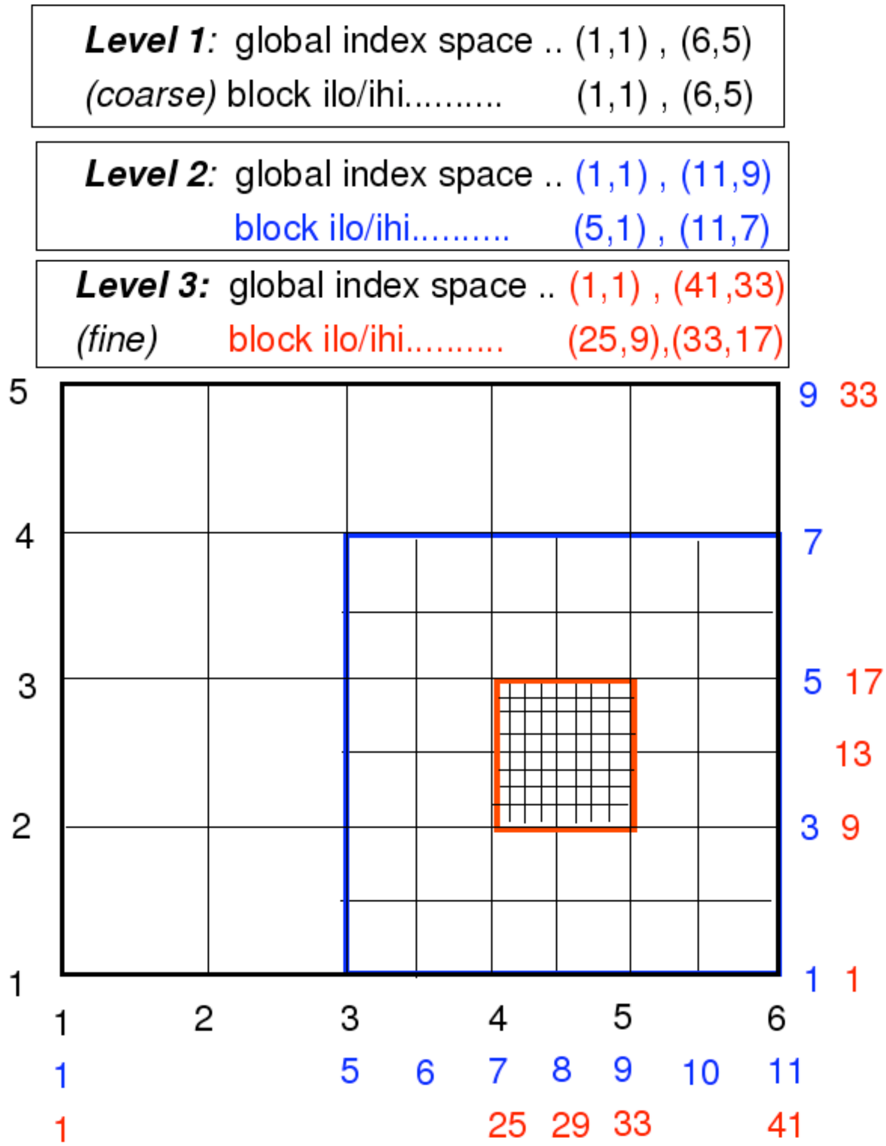

Fig. 2 - Global index space definition

From this block information the X,Y,Z location of any node `J,K,L` on the block `N` can be computed:

```
LN = LEVNO(N)
DX = (DOMXHI-DOMXLO) / ((NXC-1)*IRATIO(LN))
DY = (DOMYHI-DOMYLO) / ((NYC-1)*IRATIO(LN))
DZ = (DOMZHI-DOMZLO) / ((NZC-1)*IRATIO(LN))
X = (ILO(N,1) + J - 1) * DX
Y = (ILO(N,2) + K - 1) * DY
Z = (ILO(N,3) + L - 1) * DZ
```

The global index space makes it possible to compute the indices of overlapping points on different levels using only IRATIO:

```
J_fine = (J_coarse-1)*IRATIO(LN) + 1 !same with K,L
J_coarse = (J_fine-1)/IRATIO(LN) + 1 !same with K,L
```

which is convenient for deriving data exchanges between refinement levels for interpolations, etc. The number of fringes defines the buffer layer of data around the interior of the block.

The mesh data that resides on each block N specifies iblank and boundary information for the block. If the header information specifies block N's `IBLFLAG` parameter is 0 or 1, indicating all nodes on the block are blanked or unblanked, respectively, then the iblank data entry for the block will be null. Likewise, if the header indicates block N's `BDRYFLAG` is 0, indicating the block does not reside on a farfield boundary, the boundary data entry is null. If the iblank or boundary data entries are non-null, the format is as follows:

DATA (IBLANK):

```
IF (IBLFLAG(N) == 2) THEN
     JD = Number nodes in X
     KD = Number nodes in Y
     LD = Number nodes in Z
     IBLANK(JD,KD,LD) = iblank at each node
END IF
```

DATA (BOUNDARY):

```
IF (BDRYFLAG(N) == 1) THEN
     BXLO,BYLO,BZLO = boundary type lower faces
     BXHI,BYHI,BZHI = boundary type upper faces
END IF
```

On the boundary entry the bxlo,bxhi entries refer to the block faces (Fig. 3) with an integer value that indicates the corresponding boundary type, e.g.:

0 : interior
1 : inflow
2 : outflow
3 : periodic
…

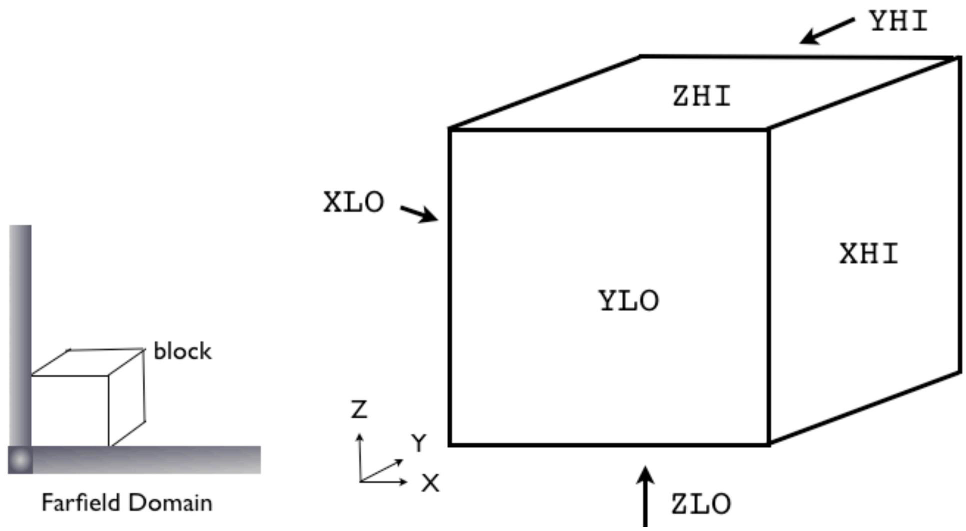

Fig. 3 – Block boundary faces