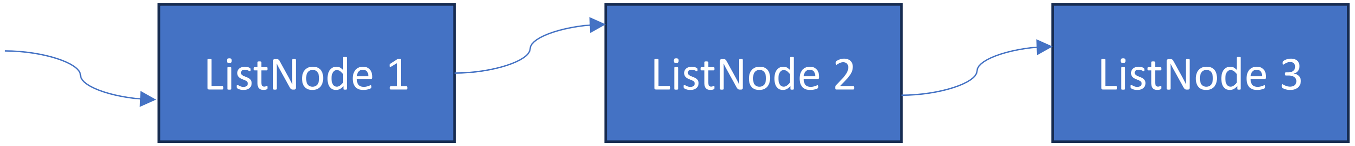
Object Serialization with HDF5

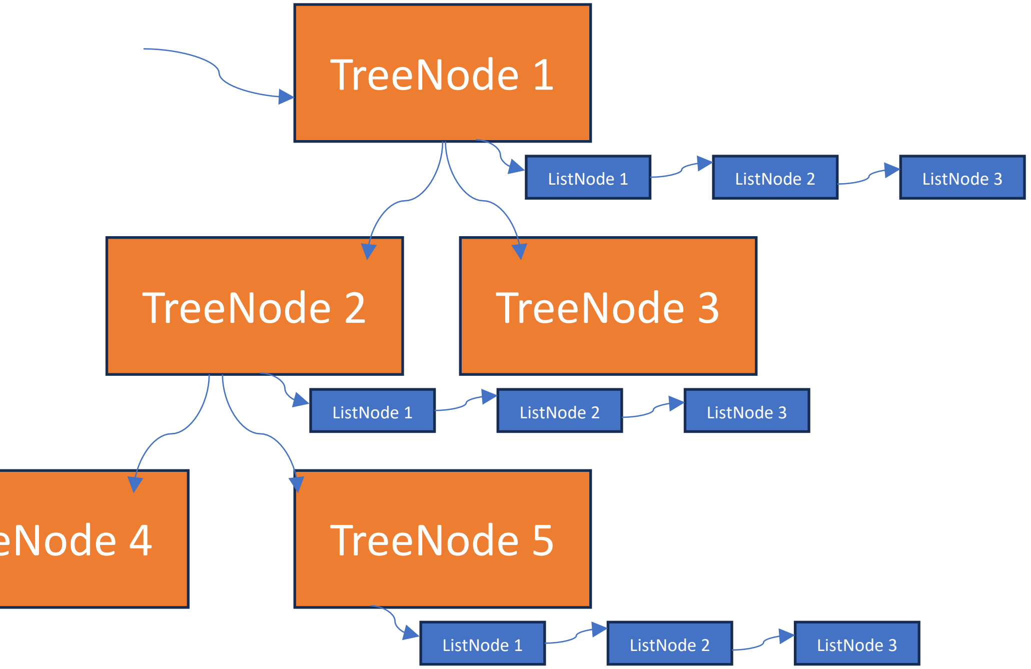
Mark C. Miller

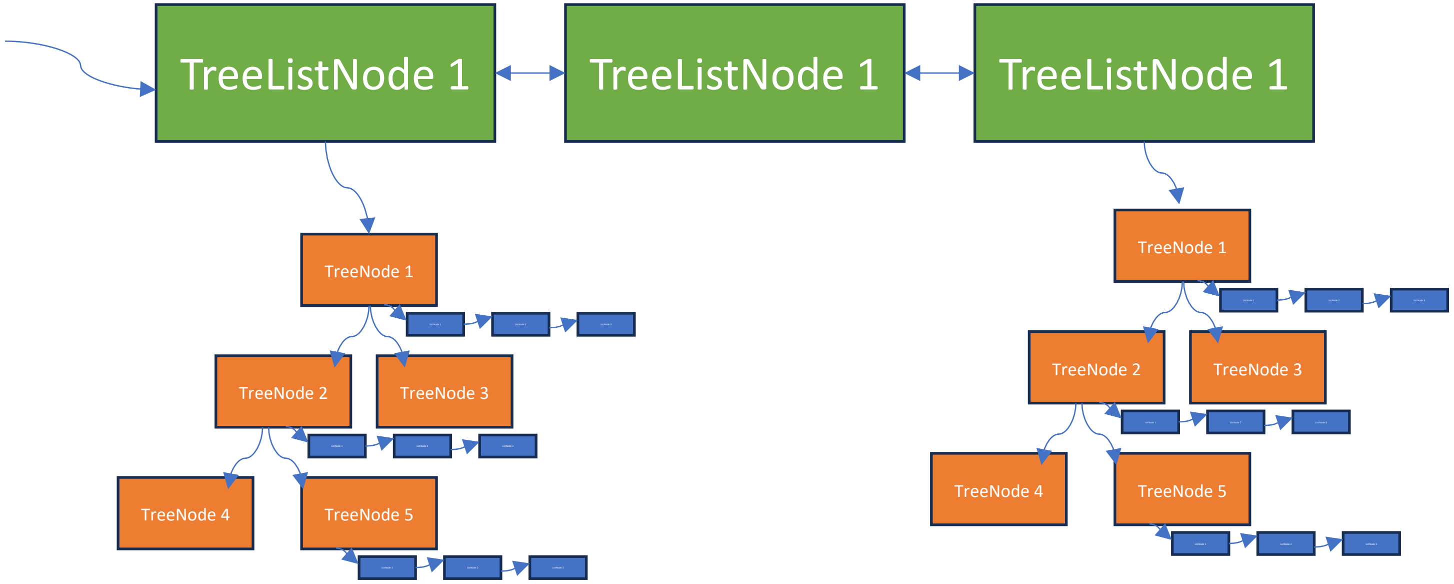
Some pointer-linked Data Structures

```
typedef struct _ListNode_t {
     int                 offsets[3];
     float               a;
     float               b;
     struct _ListNode_t *next;
} ListNode_t;

typedef struct _TreeNode_t {
    int                 memberA;
    int                 memberB;
    double              coords[3];
    struct _TreeNode_t *left;
    struct _TreeNode_t *right;
    struct _ListNode_t *list;
} TreeNode_t;

typedef struct _TreeListNode_t {
    char                    name[32];
    int                     val1;
    double                  val2;
    float                   val3;
    struct _TreeListNode_t *next;
    struct _TreeListNode_t *prev;
    struct _TreeNode_t *tree;
} TreeListNode_t;
```

Visualization of instances of these structures







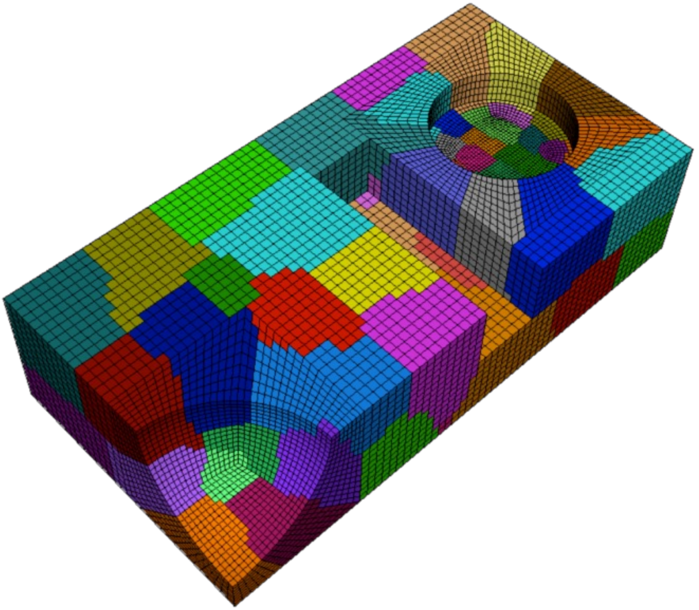
Simple coding for writing

- Define memory type(s) struct for non-pointer members of every type
    - Note: have constructed all C types such that pointers come “last” in the struct
- Recursively traverse structures
    - For every pointer, create a group and descend into it
    - For every node, create a 1 datum dataset of struct of non-pointer members
    - Write and close the dataset
- Real world example: 20 Megabytes of application data → 1.5 Gigabyte file
- But, the I/O coding is just really, really simple

Optimized approach

- Create types
  - Memory type: pointers are treated as opaque types 8 byte types
  - File type: pointers replaced by size_t offsets
  - Conversion routine to convert from opaque type to size_t
- Allocate 3 arrays of pointers, one array to point to all objects of that type
- Create 3 datasets, one for each type
- First traversal of structures…
  - fill arrays with pointers as you encounter them
- Second traversal of structures…
  - Set an H5Sselect entry in array to write and H5Dwrite that entry
  - Conversion routine converts pointer to index in dataset

Object “serialization” via File Image (core VFD)



- Goal: Applications want to migrate computational ”zones” (mesh and all its data) between MPI ranks
- Use existing I/O code used for restart to write it to memory file
- MPI send the memory file (as bytes) to target MPI rank
- Use existing I/O code to read it from memory file

https://markcmiller86.github.io/hdf5stuff/