Extending HDF5 Datasets: Enhancements to the Chunk Indexing Methods

Vailin Choi, Jerome Soumagne, Quincey Koziol
The HDF Group

July 20, 2015

Introduction

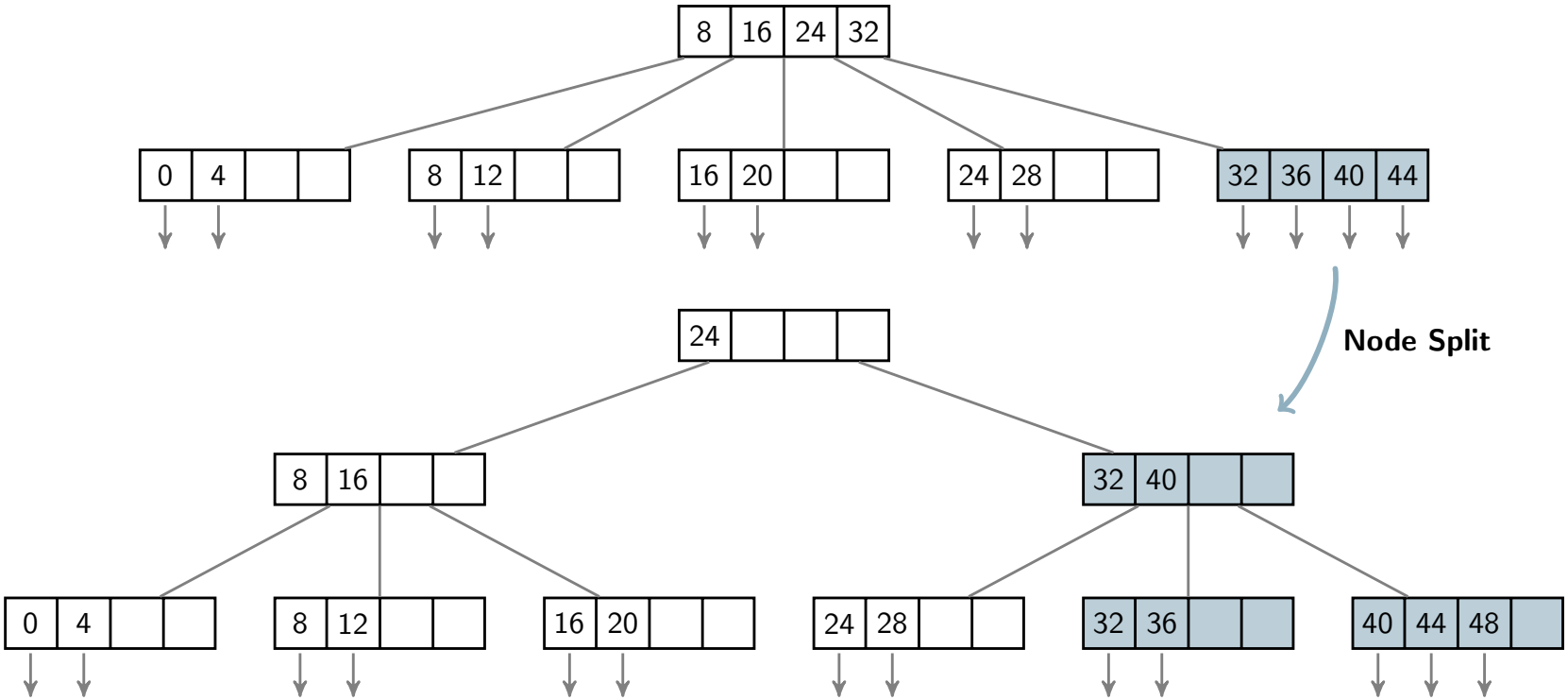
- HDF5 Chunking is used for
  - Compression
  - I/O optimization
  - **Extending datasets** efficiently
- Chunks must be indexed so that data can be retrieved efficiently
- Map of coordinates associated to chunk elements

Chunk Indexing

- Currently uses B-tree structure
- Insertion/lookup in $O(\log_b n)$, where n is the number of nodes and b the order of the tree
- Record in B-tree stores
  - Coordinates of the chunk in the dataset's dataspace.
  - The size of the chunk, in bytes.
  - The address of the chunk in the file.
  - Additional metadata.

B-Tree Issues

- Extending dataset = dataset dimension only increased at upper bound

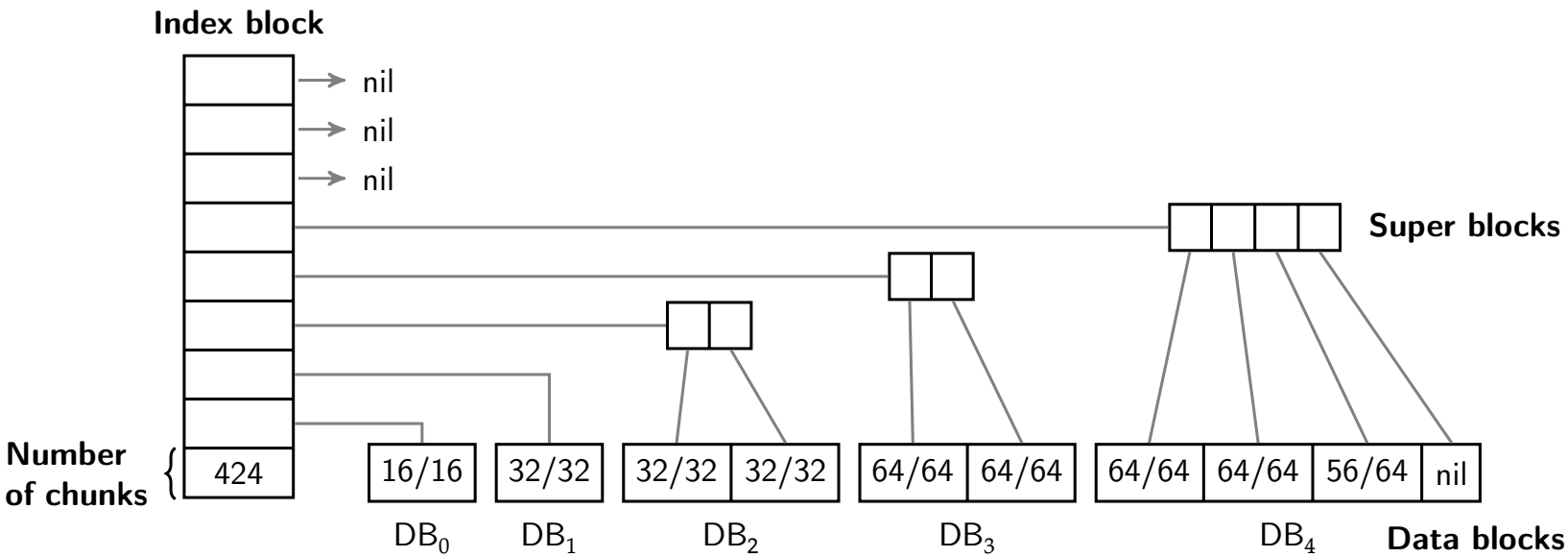


B-Tree Issues

- B-tree v2 fixes re-balancing issue
- More optimizations *tweaks* but same complexity
- B-tree not ideal for dataset extended in single dimension
  - Use extensible array instead

Extensible Array

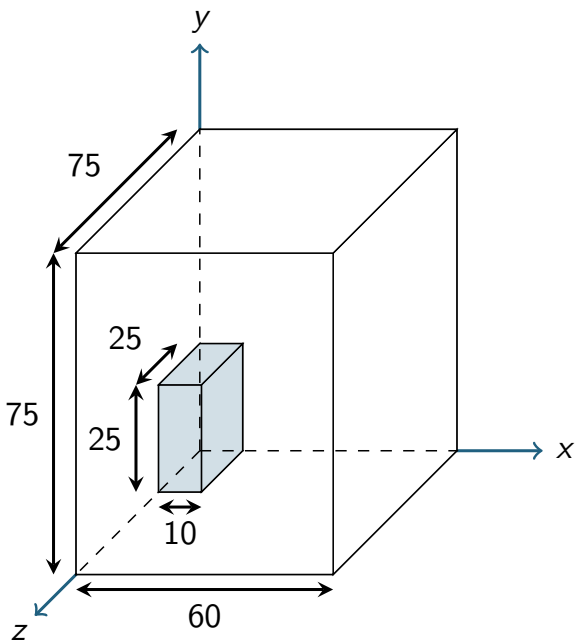
- Insertion/removal/lookup in $O(1)$



Hash Table

- Chunk present in cache are retrieved from cache hash table
- Each entry hashed based on the chunk index (varies according to dataset dimension size)
- Extending dataset means
  - Recalculation of every hash value EVERY TIME

Hashing Chunks Coordinates



Hash-value

Scaled coordinates:

$X/x = 60/10 = 6$

$Y/y = 75/25 = 3$

$Z/z = 75/25 = 3$

Number of bits to represent coordinates:

$\lceil \log_2(X/x) \rceil = 3$

$\lceil \log_2(Y/y) \rceil = 2$

$\lceil \log_2(Z/z) \rceil = 2$

0	0	0	0	0	0	0
z		y		x		

Chunk Indexing Performance

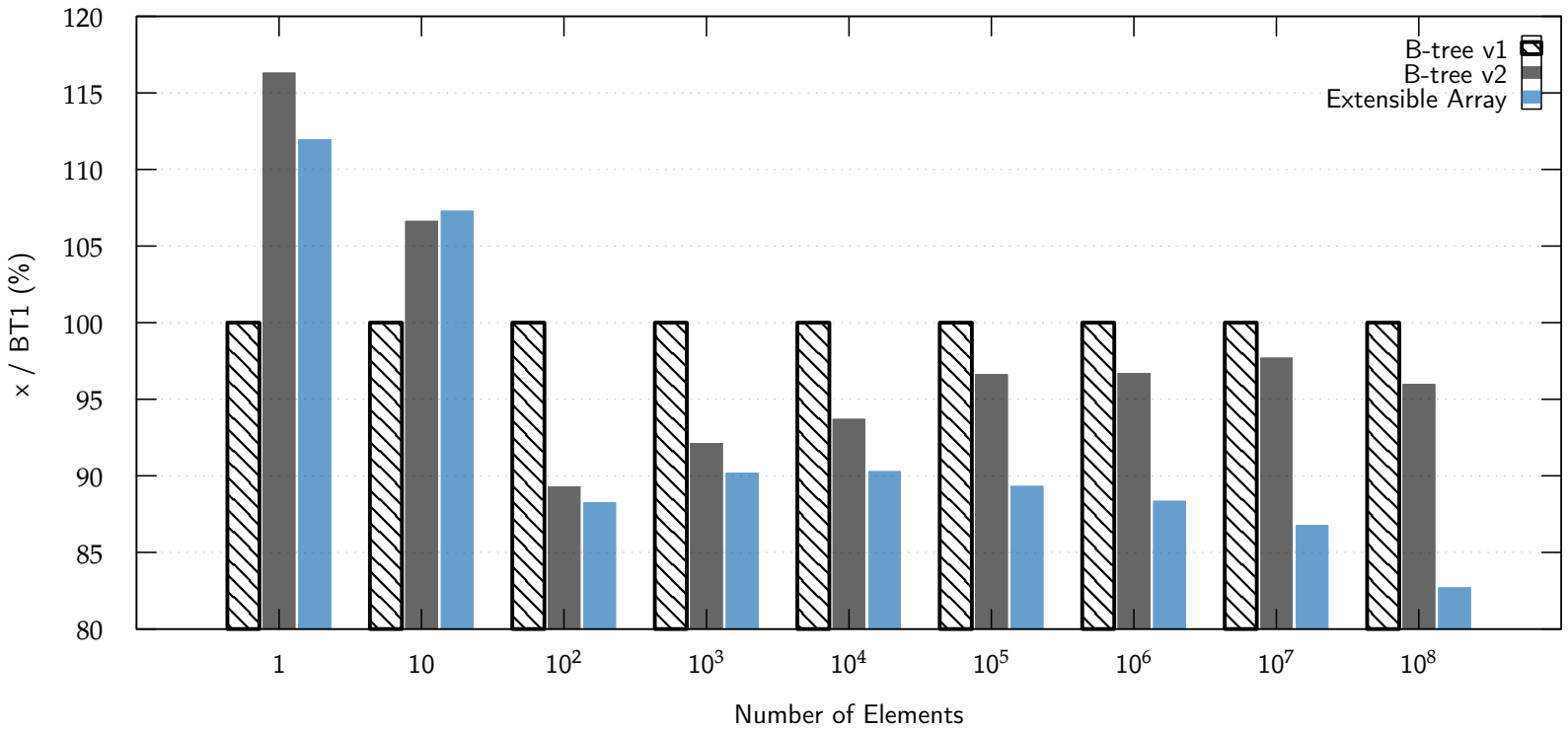
- Before

Indexing Method	Time (s)	
Along the X-direction		
EA	149.82	≃ −4%
BT1	156.15	≃ +7%
BT2	166.56	
Along the Y-direction		
EA	150.08	≃ −8%
BT1	163.82	≃ +2%
BT2	167.91	
Along the XY-direction		
EA	–	
BT1	104.08	≃ +10%
BT2	114.39	

- After

Indexing Method	Time (s)	
Along the X-direction		
EA	32.10	≃ −12.4%
BT1	36.63	≃ −3.4%
BT2	35.39	
Along the Y-direction		
EA	32.2	≃ −17.5%
BT1	39.01	≃ −10.5%
BT2	34.90	
Along the XY-direction		
EA	–	
BT1	36.11	≃ −2.3%
BT2	35.29	

Chunk Indexing Performance



Conclusion

- Obviously better performance ☺
- Paper available for more details
  - `http://svn.hdfgroup.org/hdf5doc/trunk/WhitePapers/revise_chunks/paper.pdf`
- Future work
  - Universal B-trees
  - Other structures
  - Optimize lookup of entries frequently accessed with Huffman coding

Questions