Algoritmo y Estructura de Datos

BV-Tree

Definitions
Operations
Improved methods
References

Moreno Vera Felipe Adrian
Chaparro Cruz, Israel Nazareth

Definitions

K-D Tree

Is a space-partitioning data structure. k-d tree organizes points in a k-dimensional spaces.

B Tree

Is a data structure tree that keeps data sorted and allows searches, sequential access, insertions, and deletions in logarithmic time.

K-D-B Tree

Is a k-dimensional B-Tree, The aim is to provide the search efficiency of a balanced k-d tree.

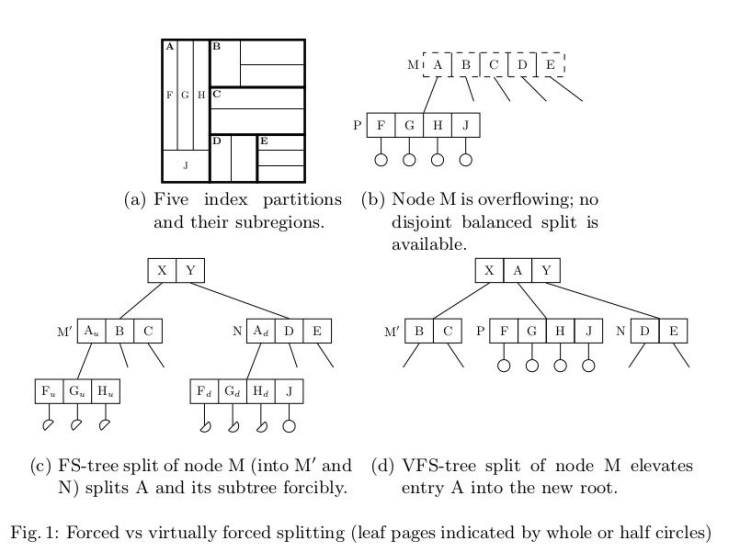
Definitions

FS-Tree

Forced Splitting Tree, is based on spatial decomposition to partition points and regions of space.

VFS-Tree

Virtual Forced Splitting Tree, is based on elevation levels like region decomposition.



(a) Five index partitions and their subregions.

(b) Node M is overflowing; no disjoint balanced split is available.

(c) FS-tree split of node M (into M′ and N) splits A and its subtree forcibly.

(d) VFS-tree split of node M elevates entry A into the new root.

Fig. 1: Forced vs virtually forced splitting (leaf pages indicated by whole or half circles)

Definitions

Fanout

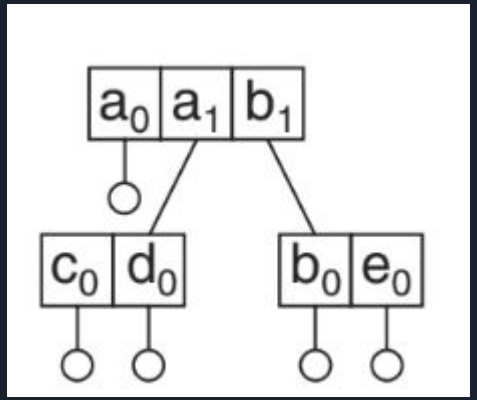
Is the number of entries which can be recorded in a single index node.

Directory nodes

Is a node that contain information (e.g. pointer) of other node or object (leaf)

Leaf node

Is a node that contain information of a specific object stored in the data structure.



Definitions

BV-Tree

Is a n-dimensional B-Tree model, also is a practical VFS-Tree, with representation as bottom-up levels model. When exceed capacity, split in at least ⅓ of data.

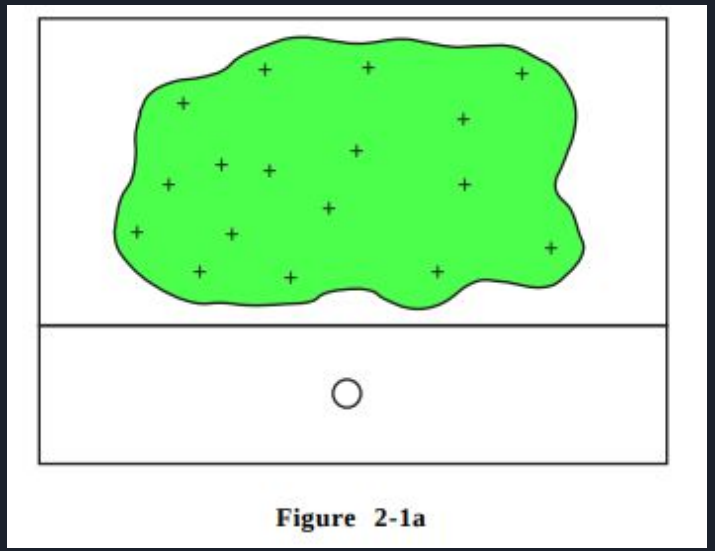


Figure 2-1a

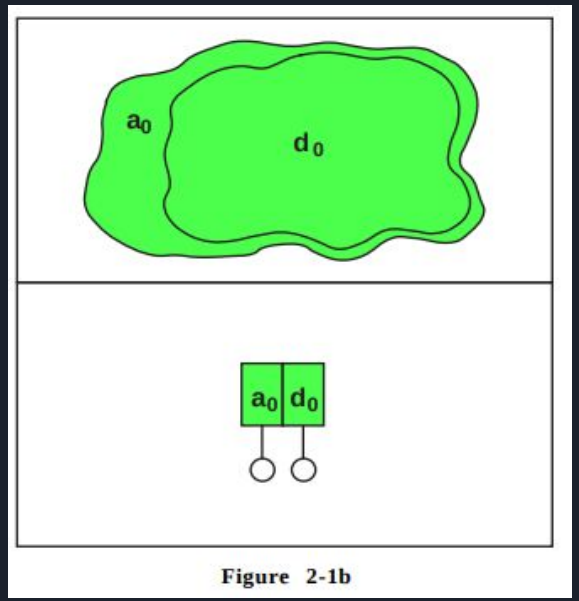


Figure 2-1b

BV-Tree

Levels: Regions and directory (hierarchy)

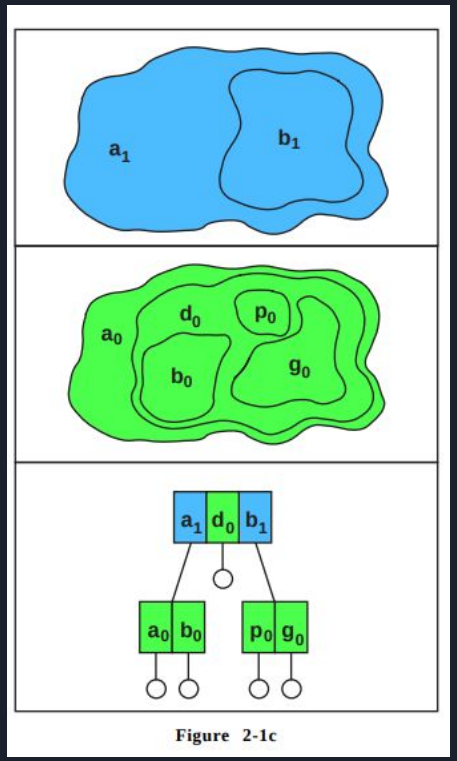


Figure 2-1c

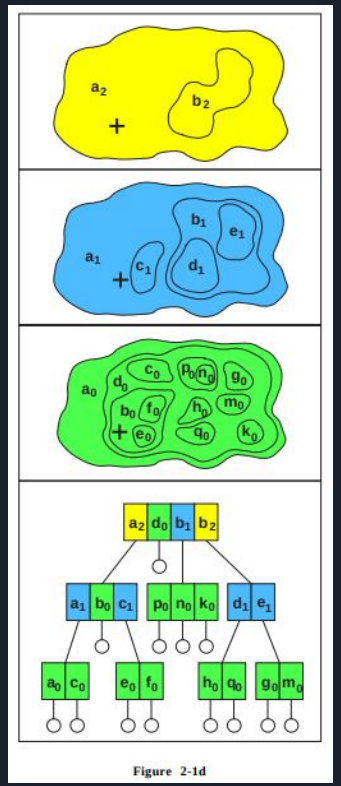


Figure 2-1d

Operations

Insert

To insert elements in BV-Tree, we use partition like VFS-Tree.
That consist in determinate when split a node, we have some cases:
* Elevation export upperbound
* Elevation import upperbound
* Over-elevation

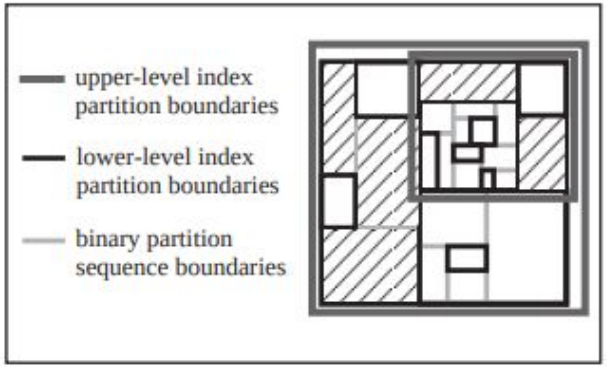


Figure 1-3: Enclosed subspaces through partition sequences

Operations

Search

To search an elements in BV-Tree, all nodes of all levels are visited to found the most closest region with contain the element.

For example, to find the “+” region in the image above, begin from root we check d0 that is the best result but it not have a next level to still search.
So, we need to search in the next region (in this case the left side of d0 region) in a2 region, we found b0 as the best result at this level and is better than d0, so we finish the search process.

Delete

To delete an elements in BV-Tree, we have two problems, first: find a region with which an underflowing region can merge and find a way of split the resulting merge region.

Improved Methods

Reduced VFS-Tree

Consist in a reduced operation that converts a full VFS-Tree into the RVFS-Tree that contain no elevated entries and are fully balanced.

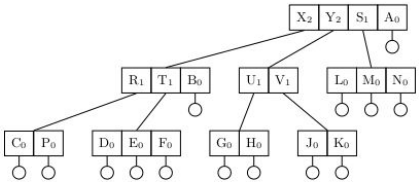


Fig. 6: VFS (BV) tree.

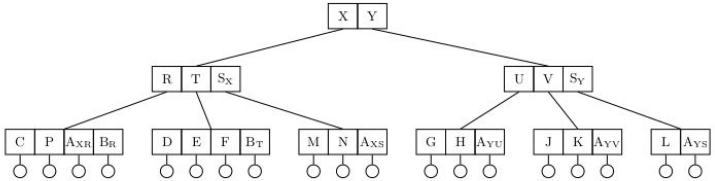


Fig. 7: Reduced VFS (BV) tree.

References

- Alan P. Sexton, Richard Swinbank, Virtual forced Splitting, demotion and the BV-Tree, In: BNCOD 2008: Sharing Data, Information and Knowledge pp 139-152.
- Michael Freeston, A General Solution of the n-dimensional B-tree Problem. In: SIGMOD Conference, San Jose, May 1995.
- Michael Freeston, On the Complexity of BV-tree Updates. In: CDB 1997: Constraint Databases and Applications pp 282-293.
- Richard Swinbank, Virtual Forced Splitting In Multidimensional Access Methods, Doctoral Thesis, The University of Birmingham.

Thanks and commend

Moreno Vera Felipe Adrian