Conclusion of Trees and B-Trees

11-4-2003

Opening Discussion

- Who can tell me what a B-tree is, when we use them, and why they are good at that purpose?
- After hours you should keep lab doors closed. Make certain they are closed and locked when you leave.
- Scheduling and courses for next semester.

Recap of B-Trees and Insertion

- As you should recall, a B-tree is a sorted tree where each node can have MANY children. It is optimized so the nodes are a full page on disk.
- All leaves are at the same level.
- Insertion is easy if nodes aren't full, but if a node is full we have to split it. And give the parent an extra key and an extra child. When the root splits we make a new root.
Deleting from a B-Tree

- The delete procedure is a bit more complex, but still fairly comprehensible. In order to do it in a single pass we put a stronger constraint on nodes as we go down at they must have $t$ keys.
- The method we use is recursive and we can break it into three cases.
  - If we find it in a leaf we simply delete it.

Delete Second Case

- If we find $k$ in an internal node we have the following cases.
  - If the child before it has at least $t$ keys then we replace $k$ with the predecessor of $k$ and delete the predecessor.
  - If the child before doesn't have that many and the child after does we do that with successor.
  - If neither adjacent child has $t$ children we merge them.

Delete Third Case

- In this case $k$ isn't in the current node. So we figure out what node it is in. If that node has $t$ or more children we simply recurse to it. Otherwise we do the following before recursing.
  - If an immediate sibling of that child has enough nodes, we move one "across from it".
  - If both immediate siblings have $t-1$ keys then we merge with one of those siblings.
**Concluding Remarks on Trees**

- While you are far from knowing everything about trees, you know enough that you should be able to pick up anything you need to learn in the future.
- Trees can be used in two main ways that are closely related.
  - Provide structure to information.
  - Repeatedly divide information into smaller sets.

**Providing Structure**

- This is what we didn’t look at much this semester, but things like directory structures or trees that represent algebraic formulas fit into this category.
- The reason we haven’t talked about these much is because they are typically very specific to applications. You have to know about a certain type of data before you can decide how to organize it.

**Repeated Division**

- We have looked at trees for searching which have nice properties for general applications that involve large amounts of data.
- Each level of the tree breaks the data up into two or more subsets in an orderly way so that we can perform operations on the data in $O(\log n)$ time. This is the real power of trees.
**Minute Essay**

- What questions do you have regarding trees in general and B-trees in particular?
- Assignment #4 is due a week from today. Do your CLR reading for next class as we move into the topic of graphs.