Data Structures and Algorithms

AVL Trees

Printable Version

Glossary

Here are some terms that are used when discussing *AVL* trees:

- **child** -- same as for binary search trees
- **parent** -- same as for binary search trees
- **leaf** -- same as for binary binary search trees
- **grandparent** -- parent of parent
- **favorite** --the child who roots the taller subtree - if the parent's balance factor is 1, the left child is the favorite -- if the parent's balance factore is -1, the right child is the favorite -- if the parent's balance factor is 0, no child is the favorite
- **linear** -- true if the parent and child are both left children or are both right children
- **rotation** -- make a child a parent and the former parent a child -- preserves binary search tree ordering

Note that terms like *favorite* and *linear* are not found in other *AVL* search tree descriptions. They are used here in order to remove leftness and rightness issues from the main *AVL* tree algorithms.

AVL properties

An *AVL* tree has the following properties:

- Leaves have a balance factor of zero
- Interior nodes have a balance factor of -1, 0, or 1

The *balance factor* marks which child is the root of the taller subtree and is computed by subtracting the height of the right child's subtree from the height of the left child's subtree. This means that an *AVL* tree is nearly as balanced as it can be with just a little bit of slop allowed. *Red-black* trees, on the other hand, allow a good deal more slop with regards to balance.

Next: Inserting into *AVL* trees