Tree-walking algorithms: Incrementally performing a postorder walk of an N-ary tree

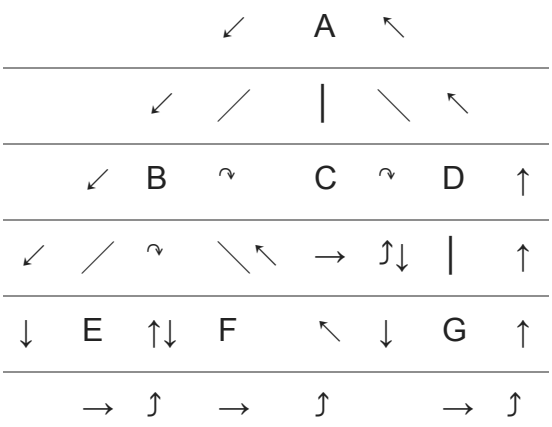
devblogs.microsoft.com/oldnewthing/20200108-00

January 8, 2020

Raymond Chen

We continue our exploration of algorithms for walking incrementally through an *N*-ary tree by perform a postorder walk through the tree.

Recall that our goal is to follow the red arrows through the tree, as if we are walking along the outside of the tree with our left hand touching it.



As with the preorder walk, we have two possible stopping places. We could stop at a leaf node (a node with no children), or we could stop at a parent node. However, the actions to take are the same in both cases, since we are always on the "upward path".

We have just finished enumerating a subtree, so we need to enumerate the next subtree, or move up to the parent if there are no more subtrees.

Therefore, we first try to move to the next sibling. If that works, then treat that subtree as a new root by going deep to the first leaf node.

If there is no next sibling, then we have finished processing all the children of a node, so move up to the parent node and enumerate it.

```
class PostorderWalker
{
  private TreeCursor cursor;

  public PostorderWalker(TreeNode node)
  {
    cursor = new TreeCursor(node);
    GoDeep();
  }

  public bool MoveNext()
  {
    if (cursor.TryMoveToNextSiblingChild()) {
      GoDeep();
      return true;
    }

    if (cursor.TryMoveToParent()) {
      return true;
    }
    return false;
  }

  public TreeNode Current => cursor.Current;

  private void GoDeep()
  {
    while (cursor.TryMoveToFirstChild()) { }
  }
}
```

Next time, we'll look at in-order enumeration.

Raymond Chen

Follow