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

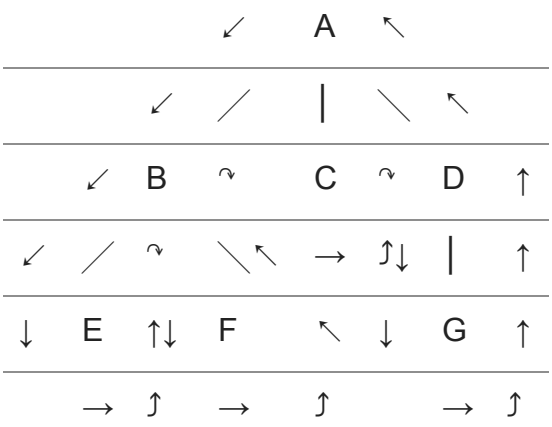
devblogs.microsoft.com/oldnewthing/20200107-00

January 7, 2020

Raymond Chen

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



Last time, we enumerated only leaf nodes, so we know that each time we resume enumeration, we are resuming from a leaf node.

This time, 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. We need to be able to resume from either.

Since we are performing a preorder walk, resuming at a parent node means that we are still on our way down the tree (walking down the left side). Therefore, we take another step down the tree.

If we resume from a leaf node, then that means we have finished walking down the tree and need to start walking back up. We have swung around to the right hand side of some subtree, and we stop moving up the tree when we find the “turn”, which happens when we find a sibling.

```
class PreorderWalker
{
  private TreeCursor cursor;

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

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

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

  public TreeNode Current => cursor.Current;
}
```

That was slightly more complicated, but not by much. We'll keep exploring next time.

Raymond Chen

Follow