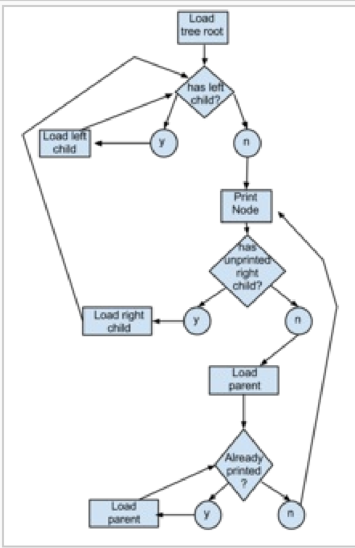
Tree sort

From Wikipedia, the free encyclopedia

This article **does not cite any references or sources**. Please help improve this article by adding citations to reliable sources. Unsourced material may be challenged and removed. *(September 2014)*

A **tree sort** is a sort algorithm that builds a binary search tree from the keys to be sorted, and then traverses the tree (in-order) so that the keys come out in sorted order. Its typical use is sorting elements adaptively: after each insertion, the set of elements seen so far is available in sorted order.

Tree sort



Class	Sorting algorithm
Data structure	Array
Worst case performance	$O(n^2)$ (unbalanced) $O(n \log n)$ (balanced)
Best case performance	$O(n \log n)$[citation needed]
Average case performance	$O(n \log n)$
Worst case space complexity	$\Theta(n)$

Contents [hide]

Efficiency [edit]

Adding one item to a binary search tree is on average an $O(\log n)$ process (in big O notation), so adding n items is an $O(n \log n)$ process, making tree sort a 'fast sort'. But adding an item to an unbalanced binary tree needs $O(n)$ time in the worst-case, when the tree resembles a linked list (degenerate tree), causing a worst case of $O(n^2)$ for this sorting algorithm. This worst case occurs when the algorithm operates on an already sorted set, or one that is nearly sorted. Expected $O(\log n)$ time can however be achieved in this case by shuffling the array.

The worst-case behaviour can be improved upon by using a self-balancing binary search tree. Using such a tree, the algorithm has an $O(n \log n)$ worst-case performance, thus being degree-optimal for a comparison sort. When using a splay tree as the binary search tree, the resulting algorithm (called splaysort) has the additional property that it is an adaptive sort, meaning that its running time is faster than $O(n \log n)$ for inputs that are nearly sorted.

Example [edit]

The following tree sort algorithm in pseudocode accepts an array of comparable items and outputs the items in ascending order:

```
STRUCTURE BinaryTree
    BinaryTree:LeftSubTree
    Object:Node
    BinaryTree:RightSubTree

PROCEDURE Insert(BinaryTree:searchTree, Object:item)
    IF searchTree.Node IS NULL THEN
        SET searchTree.Node TO item
    ELSE
        IF item IS LESS THAN searchTree.Node THEN
            Insert(searchTree.LeftSubTree, item)
        ELSE
            Insert(searchTree.RightSubTree, item)
```

```
PROCEDURE InOrder(BinaryTree:searchTree)
    IF searchTree.Node IS NULL THEN
        EXIT PROCEDURE
    ELSE
        InOrder(searchTree.LeftSubTree)
        EMIT searchTree.Node
        InOrder(searchTree.RightSubTree)

PROCEDURE TreeSort(Array:items)
    BinaryTree:searchTree

    FOR EACH individualItem IN items
        Insert(searchTree, individualItem)

    InOrder(searchTree)
```

In a simple functional programming form, the algorithm (in Haskell) would look something like this:

```haskell
data Tree a = Leaf | Node (Tree a) a (Tree a)

insert :: Ord a => a -> Tree a -> Tree a
insert x Leaf = Node Leaf x Leaf
insert x (Node t y s)
    | x <= y = Node (insert x t) y s
    | x > y  = Node t y (insert x s)

flatten :: Tree a -> [a]
flatten Leaf = []
flatten (Node t x s) = flatten t ++ [x] ++ flatten s

treesort :: Ord a => [a] -> [a]
treesort = flatten . foldr insert Leaf
```

In the above implementation, both the insertion algorithm and the retrieval algorithm have $O(n^2)$ worst-case scenarios.

See also [edit]

- Heapsort: builds a binary heap out of its input instead of a binary search tree, and can be used to sort in-place (but not adaptively).

External links [edit]

- Binary Tree Java Applet and Explanation
- Tree Sort of a Linked List
- Tree Sort in C++

The Wikibook *Algorithm Implementation* has a page on the topic of: ***Binary Tree Sort***

v· t· e **Sorting algorithms** [hide]

Theory	Computational complexity theory · Big O notation · Total order · Lists · Inplacement · Stability · Comparison sort · Adaptive sort · Sorting network · Integer sorting
Exchange sorts	Bubble sort · Cocktail sort · Odd–even sort · Comb sort · Gnome sort · Quicksort · Stooge sort · Bogosort
Selection sorts	Selection sort · Heapsort · Smoothsort · Cartesian tree sort · Tournament sort · Cycle sort
Insertion sorts	Insertion sort · Shellsort · Splaysort · **Tree sort** · Library sort · Patience sorting
Merge sorts	Merge sort · Cascade merge sort · Oscillating merge sort · Polyphase merge sort · Strand sort
Distribution sorts	American flag sort · Bead sort · Bucket sort · Burstsort · Counting sort · Pigeonhole sort · Proxmap sort · Radix sort · Flashsort
Concurrent sorts	Bitonic sorter · Batcher odd–even mergesort · Pairwise sorting network
Hybrid sorts	Block sort · Timsort · Introsort · Spreadsort · JSort
Other	Topological sorting · Pancake sorting · Spaghetti sort

Categories: Sorting algorithms