

Re: Algorithm ideas

Source: <http://coding.derkeiler.com/Archive/General/comp.programming/2004-02/1682.html>

From: CBFalconer (cbfalconer_at_yahoo.com)

Date: 02/19/04

Date: Thu, 19 Feb 2004 06:03:37 GMT

NotAsDumbAsILook wrote:

>
> *I'm updating some engineering software at work (moving it off of the
> mainframe, which entails translating the old COBOL to a Java
> application), and trying to come up with a good idea for a new
> algorithm. I'm an engineer who does programming on the side. I'm
> well-versed, but not nearly as knowledgeable as someone who pounds out
> code 40+ hours per week. I've come up with a few ideas (other than
> just copying the old algorithm, which is accurate, but slow and
> clunky). But, I'd really like to use an optimal algorithm. I want it
> to be as fast as possible. So, I'm wondering if anyone can point me to
> any books or articles that discuss algorithms that would fit my needs.
> I can't say exactly what the program is or does because of company
> proprietary information issues, but I'll make an example as best I
> can.*
>
> *Imagine you have a bunch of data that is in a tree structure. You have
> a top-level folder that contains other folders, and documents. Those
> sub-folders can, in turn, contain other sub-folders and documents. The
> structure goes down several layers deep, until a folder only contains
> documents. Each document has a number written on it. You have to add
> up all of the documents to get the value of the folder that contains
> those documents. In a folder that contains documents and sub-folders
> it's value is the sum of the documents plus the sum of the values of
> the sub-folders. This addition rolls all the way up to the top-level
> folder until it is given a value.*
>
> *One requirement is that I need to be able to look at any folder in the
> structure and see what it's value is (the sum of everything below it).
> That means I can't just parse through the tree and add up all of the
> documents. This has to be a recursive thing that gives me the
> subtotals at each folder.*
>
> *Speed is most important (next to accuracy). A user who is looking at
> the top-most folder in this structure will be dealing with 15,000+
> documents plus the sub-folders, so something on the order of 20,000
> items that the algorithm would have to crank through when updates to
> the documents are made. The user's computers really get a good*

comp.programming: Re: Algorithm ideas

- > *workout.*
- >
- > *So, if anyone has seen anything resembling this in a book or*
- > *periodical, or if you just have an idea, let me know.*

Sounds fairly simple. Is the tree binary or multiway? A single in-order walk should evaluate all nodes. 20000 odd nodes should easily fit in memory, so the update mechanism should be a matter of seconds. The problem is to form that tree in the first place, which is where I suspect your company secrets come in. The techniques of Ben Pfaffs AVL trees and red-black trees may well be applicable there.

The handling problems will have to do with the other data held in the so called documents. This whole thing sounds very much like a parts list breakdown to me. In which case the tree is definitely multiway, and a proper decision has to be made on dumping and restoring it from external media, i.e. files.

--

Chuck F (cbfalconer@yahoo.com) (cbfalconer@worldnet.att.net)
Available for consulting/temporary embedded and systems.
<<http://cbfalconer.home.att.net>> USE worldnet address!