Tuesday, October 03, 2006
3:28 PM

- Generally, types of recursive algs we'll see involve splitting a list in 2 (divide & conquer approach)

- e.g. tree-like algorithms do this, like our kd-tree (splits list in 2, recurse on each list)

alg. looks something like this

alg(list):
1. if (|list| = 1) return — $T(1) = 1$
2. split into 2 lists, L, R — $O(N)$
3. alg(L) — $T(N/2)$
4. alg(R) — $T(N/2)$

$$T(N) = 2T(N/2) + O(N) + \cancel{O(1)}$$

Tuesday, October 03, 2006
3:28 PM

- we can sub in N for O(N)

$$T(1) = 1$$
$$T(N) = 2T(N/2) + N$$

} recurrence relation (for Mergesort)

- how to solve this?
- 2 Methods: telescoping sum & back substitution
- Method 1

$$T(N) = 2T(N/2) + N$$

divide by N:

$$\frac{T(N)}{N} = \frac{2T(N/2)}{N} + 1 = \underbrace{\frac{T(N/2)}{N/2} + 1}$$

telescope

let N = N/2

$$\frac{T(N/2)}{N/2} = \frac{T(N/4)}{N/4} + 1$$

let N = N/2

$$\frac{T(N/4)}{N/4} = \frac{T(N/8)}{N/8} + 1$$

$$\vdots$$

$$\frac{T(2)}{2} = \boxed{\frac{T(1)}{1}} + 1$$

usually want to see this since we know T(1) = 1

add it all up

$$\frac{T(N)}{N} + \cancel{\frac{T(N/2)}{N/2}} + \cancel{\frac{T(N/4)}{N/4}} \cdots \cancel{\frac{T(2)}{2}} = \cancel{\frac{T(N/2)}{N/2}} + 1 + \cancel{\frac{T(N/4)}{N/4}} + 1 + \cdots + \frac{T(1)}{1} + 1$$

why do we end up with $\frac{T(1)}{1}$?

assume WLOG N is a POT (power of two)
(w/out loss of generality)

let's say N = 8

$$\frac{T(N)}{N} = \frac{T(8)}{8}$$
$$\frac{T(N/2)}{N/2} = \frac{T(4)}{4}$$
$$\frac{T(N/4)}{N/4} = \frac{T(2)}{2}$$
$$\frac{T(N/8)}{N/8} = \frac{T(1)}{1}$$

important observation:

lg N steps

$$\frac{T(N)}{N} = \frac{T(1)}{1} + 1 + 1 + 1 + \cdots 1$$
$$= T(1) + \sum_{\lg N} 1$$
$$\frac{T(N)}{N} = 1 + \lg N$$
$$T(N) = N + N\lg N \in O(N \lg N)$$

$| \quad \frac{T(N/N)}{N/N} = \frac{T(1)}{1} \qquad T(N) = N + N \lg N \in O(N \lg N)$

Method 2: Recursive back substitution

let's say $T(N) = 2T(\frac{N}{2}) + N - 1$, $T(1) = 0$

level 1: $T(N) = N - 1 + 2T(\frac{N}{2})$

level 2: $= N - 1 + 2(N/2 - 1 + 2T(\frac{N}{4}))$

$= N - 1 + N - 2 + 4T(\frac{N}{4})$

level 3: $= N-1+N-2+4(N/4 - 1 + 2T(\frac{N}{8}))$

$= N-1+N-2+N-4+8T(\frac{N}{8})$

level 4: $= N-1+N-2+N-4+8(N/8 - 1 + 2T(\frac{N}{16}))$

$= N-1+N-2+N-4+N-8+16T(\frac{N}{16})$

⋮

level k: $= \sum_{i=0}^{k-1} N - \sum_{i=0}^{k-1} 2^i + 2^k T(\frac{N}{2^k})$

$\sum_{i=0}^{N} k = k(N+1)$

$\sum_{i=0}^{N} 2^i = 2^{N+1} - 1$

$N(k-1+1) - (2^{k-1+1} - 1) + 2^k T(\frac{N}{2^k})$

$= kN - 2^k + 1 + 2^k T(\frac{N}{2^k})$

$T(1)$

let $2^k = N$ idea is: $kN - 2^k + 1 + 2^k T(\frac{N}{N})$

why we did this

if $2^k = N$

this means $k = \lg N$ $(\lg 2^k = \lg N)$

k

$T(N) = (\lg N)(N) - N + 1 + NT(1)$

$= N\lg N - N + 1 + N(0)$ since $T(1) = 0$

$= N\lg N - N + 1 \in O(N\lg N)$

Handy formulae:

$$\sum_{i=0}^{N} k = k(N+1)$$

$$\sum_{i=0}^{N} i = \frac{N(N+1)}{2}$$

$$\begin{array}{l} 0+1+2+3+4 \cdots + N \\ N + \cdots \quad 4+3+2+1 \\ \hline N(N+1)/2 \end{array}$$

$$\sum_{i=0}^{N} 2^i = 2^{N+1} - 1$$

$$\sum_{i=0}^{N} i 2^i = (N-1) 2^{N+1} + 2$$

$$\sum_{i=0}^{\lg N - 1} 2^i = N - 1$$

$$\sum_{i=0}^{\lg N - 1} N \lg \frac{N}{2^i} = \frac{N \lg^2 N}{2} + \frac{N \lg N}{2}$$

Tuesday, October 03, 2006
3:28 PM

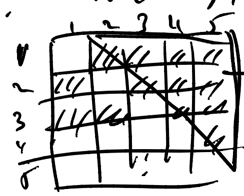
Chp 7 : Sorting

Insertion sort ; analysis : $\sum_{i=1}^{N} i = \frac{N(N+1)}{2} \in O(N^2)$

why? because it works on pairwise comparisons

How many pairs of numbers are there? $N(N-1)/2$ pairs

on average, given N numbers, with $N(N-1)/2$ pairs, $\frac{\frac{N(N-1)}{2}}{2} = \frac{N(N-1)}{4}$



$N(N-1)/2$

So... on average, insertion sort does $\frac{N(N-1)}{4}$ pairwise comparisons $\Theta\left(\frac{N(N-1)}{4}\right) \in \Theta(N^2)$

This happens to give lower bound for pairwise sort $\Omega(N^2)$ (according to text)

This includes bubble sort

Tuesday, October 03, 2006
3:28 PM

to summarize: Insertion + bubble sort : $\Omega(N^2)$
which puts them at bottom of speed pile
(they're the slowest)

Mergesort: $T(1) = 1$
$T(N) = 2T(N/2) + N$ } $O(N \lg N)$ as above

Quicksort: $T(N) = T(i) + T(N-i-1) + cN$ (from text)
$T(0) = T(1) = 1$

depends on where pivot point is chosen
worst case: pivot is at beginning, $T(N) = T(N-1) + cN$

Tuesday, October 03, 2006
3:28 PM

$$T(N) = T(N-1) + cN$$

$k=0$ $\quad = cN + T(N-1)$

$k=1$ $\quad = cN + (c(N-1) + T(N-2))$

$$= cN + cN - c + T(N-2)$$

$k=2$ $\quad = cN + cN - c + (c(N-2) + T(N-3))$

$$= cN + cN + cN - c - 2c + T(N-3)$$

$\vdots$

$$= cN(N-1) - c\sum_{k=0}^{N} k + T(N-k-1)$$

$$= c(N^2 - N) - c\frac{N(N+1)}{2} + T(N-k-1)$$

let $k=N$

$$= cN^2 - cN - \frac{cN^2 + c}{2} + \cancel{T(0)}$$

$T(N) \in O(N^2)$

$\Rightarrow$ Quicksort's worst case running time is $O(N^2)$, same as bubble or insertion

Tuesday, October 03, 2006
3:28 PM

Quicksort best case

$$T(N) = 2T\left(\frac{N}{2}\right) + cN$$

$$\vdots$$

$$= cN \lg N + N \in O(N \lg N)$$

---

summary:

bubble sort & insertion sort: $O(N^2)$

quicksort & mergesort: $O(N \lg N)$

$\downarrow$

but can degenerate to $O(N^2)$

- Best big-oh run time for sorting:

heapsort

1. construct a heap of N elements

   $O(N)$

   heap — partially sorted list with min element on top

   $\Downarrow$

   findMin() $\in O(1)$

2. perform N deleteMin() operations

   $\downarrow$

   $O(\lg N)$

   (since heap may need to be reorganized)

$O(N \lg N)$ run time, but need 2nd array (list) to store sorted values (double the memory requirements)