CMSC351 Notes

Martin Petrov

June 23, 2009

Contents

0.1 Monday June 1, 2009

0.1.1 Maximum Contiguous Subsequence

Given an array of numbers, positive and negative: $[3, -4, 2, 1, 1, 7, -5, 4, 2, -1, -3, 2, 1]$... $[2, 1, 1, 7, -5, 4, 2]$ sum to 12; any other contiguous group will be smaller.

An $O(n^3)$ solution.

M $\leftarrow 0$
for $i = 1$ to n do
  for $j = i$ to n do
    $S \leftarrow 0$
    for $k = i$ to j do
      $S \leftarrow S + A[k]$
    $M =$Max(M, S)

Return $\emptyset$ or a sum of 0 if all numbers are negative.

This number of times that $S \leftarrow S + A[k]$ is executed:

$$\sum_{i=1}^{n}\sum_{j=i}^{n}\sum_{k=i}^{j} 1 \tag{1}$$

$$\sum_{i=1}^{n}\sum_{j=i}^{n}(j - i + 1) \tag{2}$$

$$\sum_{i=1}^{n}\sum_{j=i-(i-1)}^{n-(i-1)}(j + (i-1) - i + 1) = \sum_{i=1}^{n}\sum_{j=i-(i-1)}^{n-(i-1)} j \tag{3}$$

$$\sum_{i=1}^{n}\frac{(n-i+1)(n-i+1+1)}{2} = \frac{1}{2}\sum_{i=1}^{n}(n-i+1)(n-i+2) \tag{4}$$

$$\frac{1}{2}[n(n+1) + (n-1)n + (n-2)(n-1) + ... + (3)(4) + (2)(3) + (1)(2)] \tag{5}$$

$$\frac{1}{2}\sum_{i=1}^{n} i(i+1) = \frac{1}{2}\frac{n(n+1)(n+2)}{3} \tag{6}$$

Make sure to add 1 because integers are discrete (2).
This can be solved by shifting j by $i + 1$ (3) and then using Gauss' formula (4): $\sum_{k=1}^{n} k = \frac{n(n+1)}{2}$
Expand out (5) to yield the natural summation (6).

An $O(n^2)$ solution.

M $\leftarrow 0$

for $i = 1$ to n do
    $S \leftarrow 0$
    for $j = i$ to n do
        $S \leftarrow S + A[j]$
        $M =$Max(M, S)

Find out how many times the inner region will be executed:

$$\sum_{i=1}^{n}\sum_{j=i}^{n} 1$$
$$\sum_{i=1}^{n}(n - i + 1)$$
$$n + (n-1) + (n-2) + ... + 1 = \sum_{i=1}^{n} i$$
$$\frac{n(n+1)}{2}$$

Any algorithm must run in at least linear time because any particular element might be part of the solution.

0.2 Tuesday June 2, 2009

0.2.1 Maximum Contiguous Subsequence (Continues)

An $O(n)$ solution.

M $\leftarrow 0$
S $\leftarrow 0$
for $i = 1$ to n do
    $S \leftarrow$Max$(S + A[i], 0)$
    $M =$Max(M, S)

0.2.2 Bubble Sort

M $\leftarrow 0$
for $i = n$ downto 2 do
    for $j = 1$ to $n - 1$ do
        if $A[j] > A[j + 1]$ then $A[j] \leftrightarrow A[j + 1]$

The number of comparisons:

$$\sum_{i=2}^{n}\sum_{j=1}^{i-1} 1$$
$$\sum_{i=2}^{n} i - 1$$

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

$$\sum_{i=1}^{n-1} i = \frac{n(n-1)}{n}$$

The number of swaps:
Best Case: 0
Worst Case: $\frac{n(n-1)}{n}$
Average Case: Assume each ordering is equally likely. Value of the data is irrelevant, only comparisons which give you a finite number of elements.

$$\frac{\sum \text{ all permutations } \pi_i \ C(\pi_i)}{C(\text{all permutations})}$$

Count the transpositions. In the worst case there are $\binom{n}{2}$, which equals Gauss' formula. In the best case there are zero. In this case, each pair is equally likely to be exchanged, so divide the worst case by two.

Knuth would keep track of swaps, this improves the best case runtime to $O(n)$. Cocktail shaker sort is similar and converges in the middle..

Prove Gauss' by induction

$\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$
Proof by Mathematical Induction [Need to know the answer ahead of time, strong inductive technique, weak deductive]

Base case: $n = 1 \leftrightarrow 1 = \frac{(1)(1+1)}{2}$

Deductive Step: Assume true for $n-1$ then $\sum_{i=1}^{n-1} i = \frac{(n-1)(n-1+1)}{2} = \frac{(n-1)n}{2}$

$$\sum_{i=1}^{n} i$$

$$\sum_{i=1}^{n-1} i + n$$

$$\frac{(n-1)n}{2} + n \quad \text{by IH}$$

$$\frac{n^2 - n}{2} + \frac{2n}{2}$$

$$\frac{n^2 + n}{2}$$

$$\frac{n(n+1)}{2}$$

0.3 Wedensday June 3, 2009

0.3.1 Insertion Sort

Continuously inserts an element up.

$A[0] \leftarrow -\infty$
for $i = 2$ to n do
    $temp \leftarrow A[i]$
    $j \leftarrow i - 1$
    while $temp < A[j]$ do
        $A[j+1] \leftarrow A[j]$
        $j \leftarrow j = 1$
    end while
    $A[j+1] \leftarrow temp$
end for

Comparisons

Base case: $n - 1$

Worst case:

$$\sum_{i=2}^{n} i = \sum_{i=1}^{n} i - \sum_{i=1}^{1} i = \frac{n(n+1)}{2} - 1 = \frac{n^2 + n - 2}{2}$$

Average case: Number of comparisons needed to get to j: $i - j + 1$.
Likelyhood of ending up at position j: $\frac{1}{i}$.

$$\sum_{i=2}^{n} \frac{1}{i} \sum_{j=1}^{i} i - j + 1 = \sum_{i=2}^{n} \frac{1}{i} \sum_{j=1}^{i} j =$$
$$\sum_{i=2}^{n} \frac{1}{i} \frac{i(i+1)}{2} = \sum_{i=2}^{n} \frac{i+1}{2} = \frac{1}{2} \sum_{i=3}^{n+1} i =$$
$$\frac{1}{2} \sum_{i=1}^{n+1} i = \frac{1}{2} \sum_{i=1}^{2} i = \frac{1}{2} \frac{(n+1)(n+2)}{2} - \frac{3}{2} =$$
$$\frac{n^2 + 3n - 4}{4}$$

Insertion sort w/o sentinel

for $i = 2$ to n do
    $temp \leftarrow A[i]$
    $j \leftarrow i - 1$
    while $j > 0$ and $temp < A[j]$ do
        $A[j+1] \leftarrow A[j]$
        $j \leftarrow j = 1$
    end while

$A[j+1] \leftarrow temp$
end for

Base case: $n-1$

Worst case: Number of previous comparisons minus the comparisons to the sentinel $(n-1)$.

$$\sum_{i=2}^{n} i - 1 = \sum_{i=1}^{n-1} i = \frac{n(n-1)}{2}$$

Average case: Saving $\frac{1}{i}$ per iteration. Overall savings:

$$\sum_{i=2}^{n} \frac{1}{i} = \sum_{i=1}^{n} \frac{1}{i} - \sum_{i=1}^{2} \frac{1}{i} = H(n) - 1 \approx \ln(n) - 1$$

Subtract from the worst case with the sentinel:

$$\frac{n^2+3n-4}{4} - [H(n)-1] = \frac{n(n+3)}{4} - H(n)$$

0.3.2 Selection sort

This is the best quadratic sorting algorithm available.

for $i = n$ down to 2 do
  $k \leftarrow 1$
  for $j = 2$ to i do
    if $A[j] > A[k]$ then $k \leftarrow j$
  end for
  $A[k] \leftrightarrow A[i]$
end for

Best / Worst / Average Case for Comparisons:

$$\sum_{i=2}^{n} n \sum_{j=2}^{i} 1 = \sum_{i=2}^{n} i - 1 = \sum_{i=1}^{n-1} i = \frac{n(n-1)}{2}$$

The same number of comparisons are done in all cases.

Best / Worst / Average Case for Exchanges: $n-1$ for all cases.

Inplace

Does not use very much extra storage, there is no exact definition, usually stack space is disregarded unless it's a linear amount.

0.4 Thursday June 4, 2009

0.4.1 Merge Sort

Keep splitting the list in half, sort, then merge.

```
procedure MergeSort(A, p, r)
    if r − p > 1 then
        q ← ⌊(p+r)/2⌋
        MergeSort(A, p, q)
        MergeSort(A, q + 1, r)
        Merge(A, [p, q], [q + 1, r])
    end if
end procedure
```

Merging

Merge the following two lists:
$A_1 \leq A_2 \leq A_3 \leq \ldots \leq A_n$
$B_1 \leq B_2 \leq B_3 \leq \ldots \leq B_n$

Number of comparisons in merge (worst): $2n - 1$
Number of comparisons in merge (best): n

Worst case occurs when elements are interweaved.
$A_1 \leq B_1 \leq A_2 \leq B_2 \ldots \leq A_n \leq B_n$
This is due to the number of spaces in between the elements of the merged array.

If two lists have different sizes: $m + n - 1$; however, this could be improved, if one has size one than it could be binary searched.

Merge Sort, Cont.

This algorithm is in-place theoretically, but not in practice. All work is done in the merging.
Number of comparisons: $(n-1) + 2(\frac{n}{2} - 1) + 4(\frac{n}{4} - 1) + \ldots \frac{n}{2}(1) + n(0)$

$$\sum_{i=0}^{\log_2 n-1} 2^i[\frac{n}{2^i} - 1] = \sum_{i=0}^{\log_2 n-1} (n - 2^i) =$$

$$\sum_{i=0}^{\log_2 n-1} n - \sum_{i=0}^{\log_2 n-1} 2^i = n\log_2 n - (2^{\log_2 n} - 1) = n\log_2(n) - n + 1$$

$$\begin{aligned} S &= 1 + r + r^2 + r^3 + \ldots + r^{n-1} \\ S - 1 &= r + r^2 + r^3 + \ldots + r^{n-1} \\ &= r(1 + r + r^2 + \ldots + r^{n-2}) \end{aligned}$$

$$\begin{aligned} S - 1 + r^n &= r(1 + r + r^2 + \ldots + r^{n-2} + r^{n-1}) \\ &= rS \\ S - 1 + r^n &= rS \\ S - rS &= 1 - r^n \\ S(1 - r) &= 1 - r^n \\ S &= \frac{1 - r^n}{1 - r} \text{ or if r=1} \rightarrow n \end{aligned}$$

0.5 Friday June 5, 2009

0.5.1 Order Notation

Ignore constant factors: $3n^2 = 10n^2 = \Theta(n^2)$

$\Theta(g(n))$ is a set of functions: $\{f(n)$ $\exists$ positive constants C_1, C_2, n_0 such that $0 \leq C_1 g(n) \leq f(n) \leq C_2 g(n), \forall n \geq n_0\}$

$O(g(n))$ is a set of functions: $\{f(n)$ $\exists$ positive constants C, n_0 such that $0 \leq f(n) \leq Cg(n), \forall n \geq n_0\}$

$\Omega(g(n))$ is a set of functions: $\{f(n)$ $\exists$ positive constants C, n_0 such that $0 \leq Cg(n) \leq f(n), \forall n \geq n_0\}$

$f(n) = o(g(n))$: $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$

$f(n) = \omega(g(n))$: $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$

$3n^2 + 7n = O(n^3) = \Theta(n^2) \neq \Theta(n^3)$
This can be thought of as: $\Theta \leftrightarrow =$ and $O \leftrightarrow \leq$ and $\Omega \leftrightarrow \geq$ and $o \leftrightarrow >$ and $\omega \leftrightarrow <$

```
procedure Prime(n)
    for i = 2 to n
        if i|n then return false
    end for
    return true
end procedure
```

Because $\sqrt{n}$ is the $\lim_{sup}$ and 1 is $\lim_{inf}$
$O(\sqrt{n})$, $\Theta(\sqrt{n})$, $\Omega(1)$

A Θ proof

$$\begin{aligned} 4n^2 - 2n + 3 &= \Theta(n^2) \\ f(x) &= 4n^2 - 2n + 3 \end{aligned}$$

$$
\begin{array}{cl}
g(x) = n^2 & \\
\exists \text{ positive constants } C_1, C_2, n_0 & \\
0 \leq C_1 n^2 \leq 4n^2 - 2n + 3 \leq C_2 n^2 & \forall n \geq n_0 \\
4n^2 - 2n + 3 \leq^? 4n^2 & \forall n \geq n_0 \\
3 \leq 2n & \\
n \geq \frac{2}{3} & \forall n \geq n_0 \text{ True when condition is met} \\
n_0 = 2 & \\
3n^2 \leq 4n^2 - 2n + 3 & \forall n \geq n_0 \\
3n^2 \leq 4n^2 - 2n + 3 & \forall n \geq n_0 \\
0 \leq 1n^2 - 2n + 3 & \forall n \geq n_0 \text{ Always true}
\end{array}
$$

A more exact lower bound can be found using algebra and the quadratic formula:

$$
\begin{array}{c}
0 \leq n^2 - 20n + 30 \\
0 \leq 10 \pm \sqrt{70} \approx 18.19
\end{array}
$$

0.6 Monday June 8, 2009

$$
\begin{aligned}
\frac{n^3}{6} + \frac{n^2}{2} + \frac{n}{3} &= \frac{n^3}{6} + O(n^2) \\
&= \frac{n^3}{6} + o(n^3) \\
&\sim \frac{n^3}{6}
\end{aligned}
$$

$$
\begin{array}{c}
5n^3 - 4n^2 + 3n - 1 = \Theta(n^2) \\
\exists \text{ positive constants } C_1, C_2, n_0 \\
0 \leq C_1 n^2 \leq 5n^3 - 4n^2 + 3n - 1 \leq C_2 n^3 \ \forall n \geq 0 \\
\text{let } C_2 = 5 \\
5n^3 - 4n^2 + 3n - 1 \leq 5n^3 \\
-4n^2 + 3n - 1 \leq 0 \\
4n^2 - 3n + 1 \leq 0
\end{array}
$$

This is always positive, quadratic formula yields no positive real roots, works for all n

$$
\begin{array}{c}
\text{let } C_1 = 4 \\
5n^3 - 4n^2 + 3n - 1 \geq 4n^3 \\
n^3 - 4n^2 + 3n - 1 \geq 0
\end{array}
$$

Find the derivative to check, works for somewhere $n \geq 2$

$$2n^2 - 8n + 3 \geq 0$$

True for $n \geq 4$

$$\begin{aligned}
&a_k n^k + a_{k-1}n^{k+1} + a_{k-2}n^{k+2} + \ldots + a_1 n + a_0 \\
&\leq a_k n^k + |a_{k-1}|n^{k+1} + |a_{k-2}|n^{k+2} + \ldots + |a_1|n + |a_0| \\
&\leq a_k n^k + |a_{k-1}|n^k + |a_{k-2}|n^k + \ldots + |a_1|n^k + |a_0|n^k \\
&\leq (a_k + |a_{k-1}| + |a_{k-2}| + \ldots + |a_1| + |a_0|)n^k \\
&C_1 = \sum_{i=0} k|a_0| \\
&n_0 = 0
\end{aligned}$$

We proved this is $O(n^k)$

0.7 Tuesday June 9, 2009

It takes $\Theta(n)$ to add two n-digit numbers, because one must iterate through every digit.

0.7.1 Multiplication

$$\begin{aligned}
x &= 8432 \\
y &= 6917 \\
a = 84, b = 23, c = 69, d &= 17 \\
ac * 100^2 + (ad + bc) * 10^2 + bd& \\
69 * 84 * 100^2 + (17 * 84 + 32 * 69)^2 * 10 + 17 * 32&
\end{aligned}$$

The Algorithm

procedure Mult(X,Y,n)
If $n = 1$ then return (XY)
Split X: $a(10^{\frac{n}{2}}) + b$
Split Y: $c(10^{\frac{n}{2}}) + d$
return Mult$(a, c, \frac{n}{2})$ 10^n + (Mult$(a, d, \frac{n}{2})$ +Mult$(b, c, \frac{n}{2})$) $* 10^{\frac{n}{2}}$+ Mult$(b, d, \frac{n}{2})$
end procedure

Analysis of Runtime

$$\begin{aligned}
M(n) = 4M(\frac{n}{2}) + A(n) + A(\frac{3}{2}n) \quad &\text{for } n > 1 \\
m(1) \quad &\text{for } n
\end{aligned}$$

$$A(n) = \Theta(n)$$

$$A(n) = \alpha(n)$$
$$m(n) = \mu$$

$$M(n) = 4M(\frac{n}{2}) + \frac{5}{2}\alpha n \quad \text{for } n > 1$$
$$\mu \quad \text{for } n$$

$$\begin{aligned} M(n) &= 4(M(\frac{n}{4}) + \frac{5}{2}\alpha) + \frac{5}{2}\alpha n \quad \text{for } n > 2 \\ &= 4^k M(\frac{n^k}{2}) + \sum_{j=0}^{k-1} 4^j \frac{5}{2}\alpha \frac{n}{2^j} \\ &= 4^k M(\frac{n^k}{2}) + \frac{5}{2}\alpha \sum_{j=0}^{k-1} 2^j \\ &= 4^k M(\frac{n^k}{2}) + \frac{5}{2}\alpha(2^k - 1) \\ &= 4^{\log_2 n} M(1) + \frac{5}{2}\alpha n(n-1) \\ &= n^{\log_2 4}\mu + \frac{5}{2}\alpha n(n-1) \\ &= n^2\mu + \frac{5}{2}\alpha n(n-1) \\ &= \Theta(n^2) \end{aligned}$$

In comparison, the standard algorithm is $\approx (\alpha + \mu)n^2$ asymptotically.

0.8 Wednsday June 10, 2009

Use 2^{32} as the base on computers when multiplying large numbers. This is useful for public key cryptography.

0.8.1 An Improved Multiplication Algorithm

Given numbers cd and ab We are looking for: $ac, bd, ad + bc$
$u = (ab)(cd) = ac + ad + bc + bd$
$u - (a)(c) - (b)(d)$

Example

$$x = 8432$$
$$y = 6917$$
$$a = 69, b = 17, c = 84, d = 32$$
$$u = (86)(116) = 9176 \quad \text{Obtained using a recursive call}$$

$$v = (a)(c) = (69)(84) = 5796 \quad \text{"}$$
$$w = (b)(d) = (17)(32) = 544 \quad \text{"}$$
$$u - v - w = 9176 - 5796 - 544 = 3636$$
$$57960000 + 00000544 + 363600 = 58324144$$

The Algorithm

procedure Mult(X,Y,n)
If $n = 1$ then return (XY)
Split X: $a(2^{\frac{n}{2}}) + b$
Split Y: $c(2^{\frac{n}{2}}) + d$
$u \leftarrow$Mult$(a + b, c + d, \frac{n}{2})$
$v \leftarrow$Mult$(a, c, \frac{n}{2})$
$w \leftarrow$ Mult$(b, d, \frac{n}{2})$
$z \leftarrow u - v - w$
return $v \times 2^n + z \times 2^{\frac{n}{2}} + w$
end procedure

Analysis

$$M(n) = 3M(\frac{n}{2}) + 2A(n) + 2A(n) + A(\frac{3}{2}n) \quad \text{for } n > 1 \quad \text{The recurrence relation}$$
$$= 3M(\frac{n}{2}) + \alpha n + 2\alpha n + \frac{3}{3}\alpha n$$
$$= 3M(\frac{n}{2}) + \frac{9}{2}\alpha n$$
$$= 3[3M(\frac{n}{4}) + \frac{9}{4}\alpha n] + \frac{9}{2}\alpha n$$
$$= 3[3[3M(\frac{n}{8}) + \frac{9}{8}\alpha n] + \frac{9}{4}\alpha n] + \frac{9}{2}\alpha n$$
$$= 3^4 M(\frac{n}{8}) + 3^3\frac{9}{8}\alpha n + 3^2\frac{9}{4}\alpha n + 3^1\frac{9}{2}\alpha n$$
$$= 3^k M(\frac{n}{2^k}) + \sum_{j=0}^{k-1} 3^j \frac{9}{2}\alpha\frac{n}{2^j}$$
$$= 3^k M(\frac{n}{2^k}) + \frac{9}{2}\alpha n \sum_{j=0}^{k-1}(\frac{3}{2})^j$$
$$= 3^k M(\frac{n}{2^k}) + \frac{9}{2}\alpha n \frac{(\frac{3}{2})^k - 1}{\frac{3}{2} - 1}$$
$$= 3^k M(\frac{n}{2^k}) + 9\alpha(n(\frac{3}{2})^k - 1)$$

Find k

$$\begin{aligned}\frac{n}{2^k} &= 1\\ 2^k &= 1\\ k &= \lg n\end{aligned}$$

$$\begin{aligned}&= 3^{\lg n}M(1) + 9\alpha n(\frac{3}{2})^{\lg n} - 1)\\ &= 3^{\lg n}M(1) + 9\alpha n(n^{\lg \frac{3}{2}} - 1)\\ &= n^{\lg 3}M(1) + 9\alpha n(n^{\lg 3 - \lg 2} - 1)\\ &= n^{\lg 3}M(1) + 9\alpha n(n^{\lg 3 - 1} - 1)\\ &= n^{\lg 3}M(1) + 9\alpha n^{\lg 3} - 9\alpha n\\ &= n^{\lg 3}\mu + 9\alpha n^{\lg 3} - 9\alpha n\\ &= (\mu + 9\alpha)n^{\lg 3} - 9\alpha n\end{aligned}$$

0.9 Thursday June 4, 2009

0.10 Merge Sort (Cont.)

0.10.1 The Recurrence Relation

$$\begin{aligned}S(1) &= 0\\ &= 2S(\frac{n}{2}) + M\frac{n}{2}\\ &= 2S(\frac{n}{2}) + n - 1\\ &= 2[2S(\frac{n}{4}) + n - 1] + n - 1\\ &= 2[2[2S(\frac{n}{8}) + n - 1] + n - 1] + n - 1\\ &= 2^k S(\frac{n}{2^k}) + 2^{k-1}(\frac{n}{2^{k-1}} - 1) + 2^{k-2}(\frac{n}{2^{k-2}} - 1) + \ldots + 2(\frac{n}{2} - 1) + (n - 1)\\ &= 2^k S(\frac{n}{2^k}) + \sum_{j=0}^{k-1} 2^j(\frac{n}{2^j} - 1)\\ &= 2^k S(\frac{n}{2^k}) + \sum_{j=0}^{k-1} n - 2^j\\ &= 2^k S(\frac{n}{2^k}) + \sum_{j=0}^{k-1} n - \sum_{j=0}^{k-1} 2^j\\ &= 2^k S(\frac{n}{2^k}) + kn - (2^k - 1)\\ &= nS(1) + (\lg n)n - (2^(\lg n) - 1)\\ &= 0 + (\lg n)n - (n - 1)\end{aligned}$$

$$= (\lg n)n - n + 1$$

0.10.2 General Formula for Recurrence Relation

Recurrence: cn^d branches into a copies of $c(\frac{n}{b})^d$ which branch to $c(\frac{n}{b^2})^d$ and eventually down to $c(\frac{n}{b^{k-1}})^d$. These sum up as $cn^d + ac(\frac{n}{b})^d + a^2c(\frac{n}{b^2})^d + a^{k-1}c(\frac{n}{b^{k-1}})^d + a^k f$ where $k = \log_b n$; or $\sum_{j=0}^{k-1} a^j c(\frac{n}{b^j})^d$.
If the a is large, than the very bottom row really matters, so f has the greatest weight, if a is small cn^d has a much greater weight.

$$\begin{aligned}
T(1) &= f \\
T(n) &= aT(\frac{n}{b}) + cn^d \\
&= a[aT(\frac{n}{b^2}) + c(\frac{n}{b})^d) + cb^d] \\
&= a[a[aT(\frac{n}{b^3}) + c(\frac{n}{b^2})^d)c(\frac{n}{b})^d) + cb^d] \\
&= a^kT(\frac{n}{b^k}) + a^{k-1}c(\frac{n}{b^{k-1}})^d + a^{k-2}c(\frac{n}{b^{k-2}})^d + \ldots + cn^d \\
&= a^kT(\frac{n}{b^k}) + \sum_{j=0}^{k-1} a^j c(\frac{n}{b^j})^d \\
&= a^kT(\frac{n}{b^k}) + cn^d \sum_{j=0}^{k-1} (\frac{a^j}{b^j})^d \\
&= a^kT(\frac{n}{b^k}) + cn^d \sum_{j=0}^{k-1} (\frac{a}{b^d})^j \\
&\quad \text{if } \tfrac{a}{b^d} = 1 \text{ then } a^kT(\tfrac{n}{b^k}) + cn^dk = fn^{\log_b a} + cn^d \log_b n \\
&= a^kT(\frac{n}{b^k}) + cn^d \frac{(\frac{a}{b^d})^k - 1}{\frac{a}{b^d} - 1}
\end{aligned}$$

$$\begin{aligned}
\frac{n}{b^k} &= 1 \\
b^k &= n \\
k &= \log_b n
\end{aligned}$$

$$\begin{aligned}
T(n) &= a^{\log_b n}T\frac{n}{b^{\log_b n}} + cn^d \frac{(\frac{a}{b^d})^{\log_b n} - 1}{\frac{a}{b^d} - 1} \\
T(n) &= n^{\log_b a}T(1) + cn^d \frac{n^{\log_b(\frac{a}{b^d})} - 1}{\frac{a}{b^d} - 1} \\
T(n) &= n^{\log_b a}f + cn^d \frac{n^{\log_b a - \log_b b^d} - 1}{\frac{a}{b^d} - 1}
\end{aligned}$$

$$\begin{aligned} T(n) &= n^{\log_b a} f + cn^d \frac{n^{\log_b a - d} - 1}{\frac{a}{b^d} - 1} \\ T(n) &= n^{\log_b a} f + c\frac{n^{\log_b a} - 1}{\frac{a}{b^d} - 1} \\ T(n) &= (f + \frac{c}{\frac{a}{b^d} - 1})n^{\log_b a} - \frac{cn^d}{\frac{a}{b^d} - 1} \end{aligned}$$

0.10.3 Multiplication Example

$$\begin{aligned} M(n) &= M(\frac{n}{2}) + \frac{5}{2}\alpha n \\ a &= 4 \\ b &= 2 \\ c &= \frac{5}{2}\alpha \\ c &= 1 \\ f &= \mu \end{aligned}$$

$$\frac{a}{b^d} = 2$$

$$T(n) = (f + \frac{c}{\frac{a}{b^d} - 1})n^{\log_b a} - \frac{cn^d}{\frac{a}{b^d} - 1}$$

Plug and chug

0.10.4 Merge Sort Example

The $\sum_{j=0}^{k-1} a^j c(\frac{n}{b^j})^d$ occur in two different forms (n and -1), however, the f only occurs once.

$$S(n) = 2S(\frac{n}{2}) + n - 1 = [2S(\frac{n}{2}) + n] + [2S(\frac{n}{2}) - 1] + fn^{\log ba} = n \lg n - n + 1 + 0 = nlgn - n + 1$$

$$\begin{aligned} &2S(\frac{n}{2}) + n \\ &a = 2 \\ &b = 2 \\ &c = 1 \\ &c = 1 \end{aligned}$$

Use special case

$$1n^1 \log_2 n = n \lg n$$

0.11 Friday June 12, 2009

Mathematical induction is great for proving something we already know, but quite useless when the answer is not known.

0.12 Induction

0.12.1 Find Gauss

One must have the form of the answer at least.
$\sum_{i=1}^{n} i = an^2 + bn + c$

Base case: $n = 1$, $a(1)^2 + b(1) + c = 1 = a + b + c$

Inductive Hypothesis: Assume true for $n - 1$
$\sum_{i=1}^{n-1} i = a(n-1)^2 + b(n-1) + c$

Inductive Step:

$$\begin{aligned}\sum_{i=1}^{n} i &= n + \sum_{i=1}^{n-1} i \\ &= n + a(n-1)^2 + b(n-1) + c \text{ (By IH)} \\ &= \frac{1}{2}n^2 + \frac{1}{2}n\end{aligned}$$

$$\begin{aligned}-2a + b + 1 &= b \\ a - b + c &= c \\ a &= \frac{1}{2} \\ a &= b \\ b &= \frac{1}{2}\end{aligned}$$

0.12.2 Alternate

Base case: $n = 1$, $a(1)^2 \leq 1$

Inductive Hypothesis: Assume true for $n - 1$
$\sum_{i=1}^{n-1} i \leq a(n-2)^2$

Inductive Step:

$$\begin{aligned}\sum_{i=1}^{n} i &= n + \sum_{i=1}^{n-1} i \\ &\leq n + a(n-1)^2 + a(n-2)^2 \text{ (By IH)} \\ &\leq an^2\end{aligned}$$

0.12.3 Fibonacci

$F(n) = F(n-1) + F(n-2)$

Base: $F(0) = F(1) = 1$
The sequence appears very similar to doubling which would be $F(n) = 2F(n-1)$

0.12.4 Prove Doubling

$G(1) = 1$, $G(n) = 2^{n-1}$
For this proof we'll assume the function is exponential.
$G(n) \leq ab^n$

Base Case: $G(1) = 1 \leq ab^1 = ab$

Inductive Hypothesis: $G(n-1) \leq ab^{n-1}$

Inductive Step:

$$\begin{aligned} G(n) &= 2G(n-1) \\ &\leq 2ab^{n-1} \\ &\leq 2\frac{a}{b}b^n \\ &\leq ab^n \\ &\quad \text{Pick } a = \tfrac{1}{2} \text{ and } b = 2 \end{aligned}$$

0.12.5 Prove Fibonacci

$F(n) \leq ab^n$

Base Works for F(1) and F(0)

IH $F(n-1) \leq ab^{n-1}$ and $F(n-2) \leq ab^{n-2}$

$$\begin{aligned} F(n) &= F(n-1) + F(n-2) && \text{By definition} \\ F(n) &\leq ab^{n-1} + ab^{n-2} && \text{By IH twice} \\ &\leq^{?} ab^n \end{aligned}$$

$$\begin{aligned} ab^{n-1} + ab^{n-1} &\leq ab^n \\ b^{n-1} + bn - 2 &\leq b^n \\ b + 1 &\leq b^2 \\ b^2 - b - 1 &\geq 0 \end{aligned}$$

$$b \geq \frac{1 \pm \sqrt{5}}{2}$$
$$b \geq \varphi$$

a must be ≥ 1 in order to satisfy the base case To check the approximation flip the $\geq$ and $\leq$ signs and repeat the proof. Because it comes out to φ we can show that this is an excellent approximation. This gives us the answer within a factor of the golden ratio.
$(\frac{1\pm\sqrt{5}}{2})^{n-1} \leq F(n) \leq (\frac{1\pm\sqrt{5}}{2})^{n}$

0.13 Monday June 15, 2009

Heap is a binary tree where largest element is at the root, the two children of the root are both heaps. This is similar to selection sort, finding the maximum.
Preferably the tree is as balanced as possible.

Height of a binary tree: $\lg n$ (worst case)
Two comparisons per level, one of the two children, and one with the element of the larger child.
As a rough approximation this is $\approx 2n \lg n$

A better approximation is $\sum_{i=1}^{n} 2 \lg i$ because the tree is shrinking.

$$\sum_{i=1}^{n} 2 \lg i \tag{7}$$
$$2 \sum_{i=1}^{n} \lg i \tag{8}$$
$$2[\lg 1 + \lg 2 + \lg 3 + \ldots + \lg(n-1) + \lg n] \tag{9}$$
$$2 \lg(n!) \tag{10}$$
$$2 \lg[(\frac{n}{e})^n \sqrt{2\pi n}] \tag{11}$$
$$2[\lg n^n - \lg e^n + \lg \sqrt{n} + \lg \sqrt{2\pi}] \tag{12}$$
$$2[2 \lg n - n \lg e + \frac{1}{2} \lg n + \lg \sqrt{2pi}] \tag{13}$$
$$2n \lg n + O(n) \tag{14}$$

To solve this use Sterling's formula $n! \approx (\frac{n}{e})^n \sqrt{2\pi n}$ in (10).

```
procedure HeapSort(A,n)
    CreateHeap(n)
    for i = n downto 1
        Remove top element, set this as A[n]
        Return heap
    end for
end procedure
```

To create the heap, heapify starting from the bottom and work your way up.
The efficiency of the heap creation is the number of leaves per level *times* the number of comparisons needed.

$$\begin{aligned}
&= \frac{n}{2}0 + \frac{n}{4}2 + \frac{n}{8}4 + \frac{n}{16}6 + \frac{n}{32}8 + \frac{n}{64}10 + \dots \\
&= n[\frac{1}{2} + \frac{2}{4} + \frac{3}{8} + \frac{4}{16} + \frac{5}{32} + \dots] \\
&= n[\frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{16} + \frac{1}{32} + \dots + \\
&\quad \frac{1}{4} + \frac{1}{8} + \frac{1}{16} + \frac{1}{32} + \dots + \\
&\quad \frac{1}{8} + \frac{1}{16} + \frac{1}{32} + \dots +] \\
&= n[1 + \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \dots] \\
&\leq 2n
\end{aligned}$$

0.14 Tuesday June 16, 2009

The tree itself is stored as an array, with the root being at first index. Every element's left child is double its index, add one more to get the right. The parent can be found by halving and flooring.

Creating the heap: $2n + O(1)$
Comparisons: $2n \lg n + O(n)$

```
procedure HeapSort(A, n)
    for i = n/2 downto 1 do
        Sift(i, n)
     end for
    for i = n downto 2 do
        A[i] ↔ A[1]
        Sift(i, i − 1)
    end for
end procedure
```

Merge Sort is NOT in place.
Heap sort is $2n \lg n + O(n)$, whereas Merge Sort is $n \lg n + O(n)$
Heap sort is NOT stable.
Merge Sort has more overhead due to recursion?

Elements tend to sift down to the bottom because they originate down from the bottom. Half the elements in the heap are always at the bottom. Average distance from the bottom: $\frac{1}{2}0 + \frac{1}{4}1 + \frac{1}{8}2 + \frac{1}{16}3 = 1$ level
By sifting down without comparing, and then sifting up the removed element when it reaches the lowest level it will sift up an average of one time (Floyd's version).

If one wants to get a good worst case one can record the path and then binary search it, this is not practical, but would yield $\lg \lg n$ for sifting.

0.15 Wednesday June 17, 2009

0.15.1 Quick Sort

Pick a pivot, move everyone to the right sides relative to that person

```
procedure QuickSort(a, p, r)
    if p < r then
        q ← Partition(A, p, r)
        QuickSort(A, p, q − 1)
        QuickSort(A, q + 1, r)
end procedure
```

$n - 1$ comparisons per partition.

```
procedure Partition(A, p, r)
    x ← A[r]
    i ← p − 1
    for j ← p to r − 1 do
        if A[j] ≤ x then
            i ← i + 1
            A[i] ↔ A[j]
    A[i + 1] ↔ A[r]
    return i + 1
end procedure
```

Analysis

Worst case occurs when sorted, in this case it does a comparison with every element to every other element. In this case the number of comparisons is $\sum_{i=1}^{n-1} = \frac{n(n-1)}{2}$.
In the best case its $T(n) = 2T(\frac{n}{2}) + n - 1$, $T(0) = T(1) = 0$, works if it is a power of 2 minus 1. This equals $n \lg n - n + 1$

In the average case it is likely to be located either $\frac{1}{4}$ or $\frac{3}{4}$ of the way through.
This can be rewritten as the recurrence relation: $T(n) = T(\frac{3}{4}) + T(\frac{1}{4}) + n - 1$. This would be $O(n \lg n)$.

0.16 Thursday June 18, 2009

A heap may be used to implement a very efficient priority queue.

0.16.1 Quick Sort, Prove using constructive induction:

$T(n) = T(\frac{3}{4}) + T(\frac{1}{4}) + n - 1 \leq^? an \lg n$

Inductive Hypothesis

for $k < n \rightarrow T(k) \leq ak \lg n$

Inductive Step

$$\begin{aligned} T(n) &= T(\frac{3}{4}) + T(\frac{1}{4}) + n - 1 \\ &\leq a(\frac{n}{4}) \lg \frac{n}{4} + a(\frac{3n}{4}) \lg \frac{3n}{4} + n - 1 \text{ (By IH twice)} \\ &\leq a(\frac{n}{4})(\lg n - \lg 4) + a(\frac{3n}{4}) \lg(\lg n - \lg \frac{3}{4}) + n - 1 \\ &\leq an \lg n + [-\frac{a}{4} \lg 4 - a\frac{3}{4} \lg \frac{3}{4} + 1]n - 1 \\ &\leq an \lg n + [-2a + \frac{3}{4} \lg(3)a + 1]n - 1 \end{aligned}$$

We need this to be zero or less:

$$\begin{aligned} -2a + \frac{3}{4} \lg(3)a + 1 &\leq^? 0 \\ a(2 - \frac{3}{4} \lg(3)) &\geq^? 1 \\ a \geq \frac{1}{2 - \frac{3}{4} \lg(3)} &\approx 1.23 \end{aligned}$$

This appears to optimistic because it deteriorates too quickly in the worst case.

This can be better analysed as: $\sum_{\text{all pairs } x,\, y} p(x, y)$
The smallest and largest can only be compared if one is picked as the pivot in the first step. The probability of this is $\frac{2}{n}$.

$a_1 \leq a_2 \leq a_3 \leq \ldots \leq a_n$
Chance that a_j or a_i is the pivot: $\frac{2}{j-i+1}$, this is the only time they are compared.

$\sum_{\text{all pairs } x,\, y} (\frac{2}{j-i+1})$

$$\sum_{1 \leq i \leq j \leq n} (\frac{2}{j - i + 1})$$

$$\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} (\frac{2}{j - i + 1})$$

$$\sum_{i=1}^{n-1}\sum_{j=2}^{n-i-1}(\frac{2}{j})$$
$$\sum_{i=1}^{n-1}2[\sum_{j=1}^{n-i-1}(\frac{1}{j})-1]$$
$$2\sum_{i=1}^{n-1}[\ln n-i-1+\gamma)-1]$$
$$2\sum_{i=1}^{n-1}[\ln n-i-1]+2\sum_{i=1}^{n-1}+[\gamma)-1]$$
$$2\ln(n!)+O(n)$$
$$\approx 2\ln[(\frac{n}{e})^n\sqrt{2\pi n}]+O(n)$$
$$\approx 2n\ln n+O(n)=2n\frac{\lg n}{\lg e}\approx 1.38n\lg n$$

0.17 Thursday June 19, 2009

[1]

$$\sum_{i=1}^{n}\lg x_i=\lg\prod_{i=1}^{n}x_i$$

Telescoping:

$$\sum_{i=1}^{n}\frac{1}{k(k+1)}=(1-\frac{1}{2})+(\frac{1}{2}-\frac{1}{3})+(\frac{1}{n-1}-\frac{1}{n})=1-\frac{1}{n}$$

$$\sum_{i=1}^{n}k\leq\sum_{i=1}^{n}n=n^2$$

$$\sum_{i=1}^{n}k=\sum_{i=1}^{\frac{n}{2}}k+\sum_{i=1+\frac{n}{2}}^{n}k=(\frac{n}{2})^2+\frac{3n^2}{4}$$

$$\sum_{i=1}^{n}k\geq\sum_{i=1}^{1}n=n$$

$$\sum_{i=1}^{n}\frac{k^2}{2^k}$$

$$\ldots+\frac{k^2}{2^k}+\frac{(k+1)^2}{2^{k+1}}+\ldots$$

[1] Look at Appendix A of the book

Find the ratio if two adjacent terms

$$\frac{\frac{k^2}{2^k}}{\frac{(k+1)^2}{2^{k+1}}} = \frac{(1+\frac{1}{k})^2}{2}$$

$k = 1 : 2$
$k = 2 : \frac{9}{8}$
$k = 3 : \frac{8}{9}$
$k = 4 : \frac{25}{32}$

$$\sum_{i=1}^{n} \frac{k^2}{2^k} = \sum_{i=1}^{2} \frac{k^2}{2^k} + \sum_{i=3}^{\infty} \frac{k^2}{2^k} = \frac{1}{2} + 1 + \frac{9}{8}\sum_{k=0}^{\infty}(\frac{8}{9})^k = \frac{1}{2} + 1 + \frac{9}{8} + \frac{1}{1-\frac{25}{35}} \approx 7$$

Harmonic

$$\begin{aligned} H_n &= 1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5} + \frac{1}{6} + \frac{1}{7} + \dots \\ &\leq 1 + \frac{1}{2} + \frac{1}{2} + \frac{1}{4} + \frac{1}{4} + \frac{1}{4} + \frac{1}{4} + \dots \\ &\leq 1 + 1 + 1 + \dots \\ &k \approx \lg(n+1) \end{aligned}$$

Monotonically decreasing.

$$\sum_{i=1}^{n} f(k) \leq \int_{m}^{n-1} f(x)dx$$

$$\int_1^n x dx \leq \sum_{i=1}^{n} k \leq \int_1^{n+1} x dx$$
$$\frac{n^2}{2} \leq \sum_{i=1}^{n} k \leq \frac{n(n+2)}{2}$$

0.18 Monday June 22, 2009

$$\int_0^{n+1} \frac{1}{x}dx \leq \sum_{n}^{k=1} \frac{1}{x} \leq \int_0^n \frac{1}{x}dx$$
$$\int_0^{n+1} \frac{1}{x}dx \leq \sum_{n}^{k=1} \frac{1}{x} \leq 1 + \sum_{n}^{k=2} \frac{1}{x} \leq 1 + \int_1^n \frac{1}{x}dx = 1 + \ln n$$

0.18.1 Quick Sort

$$T(n) = T(q-1) + T(n-q) + n - 1$$

Each q is equally likely: $\frac{1}{n}$

$$T(n) = \sum_{q=1}^{n} \frac{1}{n}[T(q-1) + T(n-q)] + n - 1$$

$$T(1) = T(0) = 0$$

$$T(n) = \frac{1}{n}[\sum_{q=1}^{n} T(q-1) + \sum_{q=1}^{n} T(n-q)] + n - 1$$

Both of the summations expand to:

$$T(0) + T(1) + T(2) + \ldots + T(n-1)$$

$$T(n-1) + \ldots + T(0)$$

... so they can be merged

$$T(n) = \frac{2}{n} \sum_{q=0}^{n-1} T(q) + n - 1$$

Guess $T(n) \leq an \lg n$ for $n \geq 0$, Strong Induction Time!

$$\frac{2}{n} \sum_{q=0}^{n-1} T(q) + n - 1 = \frac{2}{n} \sum_{q=0}^{n-1} T(q) + n - 1$$

$$\frac{2}{n} \sum_{q=0}^{n-1} T(q) + n - 1 \leq \frac{2}{n} \sum_{q=0}^{n-1} [aq \lg q] + n - 1 \text{ (By IH } n-1 \text{ times)}$$

$$\leq \frac{2a}{n} \sum_{q=0}^{n-1} [q \lg q] + n - 1$$

Write it out...

$$1 \ln 1 + 2 \ln 2 + 3 \ln 3 + (n-1) \ln (n-1)$$

$$\leq \frac{2a}{n} \int_1^n [xlnx] dx + n - 1$$

$$\leq \frac{2a}{n} [\frac{x^2 \ln x}{2} - \frac{x^2}{4}]|_1^n + n - 1$$

$$\leq \frac{2a}{n} \frac{n^2 \ln n}{2} - \frac{n^2}{4} + \frac{1}{4} + n - 1$$

$$\leq an \ln n - \frac{an}{2} + \frac{a}{2n} + n - 1$$

$$\leq an \ln n - (1 - \frac{a}{2})n + \frac{a}{2n} - 1$$

$$\leq an \ln n$$

This is true when $1 - \frac{a}{2} \leq 0$, thus $a \geq 2$

Quick sort is $T(n) \leq 2n \ln n = 1.38n \lg n$
In practice do not recurse to bottom, stop somewhere and use an alternate sort method.

The stack size can grow to a quadratic amount, in the average case this should be roughly logarithmic.
The stack space can be improved by quick sorting on the smaller one first.

```
procedure QuickSort(a, p, r)
    if p < r then
        q ← Partition(A, p, r)
        if q − 1 − p ≤ r − (q + 1)
            QuickSort(A, p, q − 1)
            QuickSort(A, q + 1, r)
        else
            QuickSort(A, q + 1, r)
            QuickSort(A, p, q − 1)
end procedure
```

In practice this does not need to be done as it ends up logarithmic.
The last statement in the procedure can always be avoided (Tail recursion).

```
procedure QuickSort(a, p, r)
    while r = p > 1 then
        q ← Partition(A, p, r)
            QuickSort(A, p, q − 1)
        p ← q + 1
 end procedure
```

One can shuffle the elements of the array in a random permutation.
To improve this pick an i, interchange the element at i and the pivot location.
One may also randomly pick 3 elements and partition on their median, however, the effectiveness diminishes quickly.

Merge Sort $= n \lg n$, 'not in place', stable
Heap Sort $= n \lg n$, in place, not stable
Quick Sort $= 1.38n \lg n$ (average case), in place, not stable