# Divide and Conquer

EECS 477

Lecture 14, 10/31/2002

## Divide and Conquer

```
result_t algoDC(x) {
  if(x is small) return algoAdHoc(x);
  decompose x into x[0], x[1], …, x[n-1]
  for(i=0; i<n; ++i) y[i]=algoDC(x[i]);
  recombine y[i] to obtain solution y
  return y;
}
```

## Analyze via recurrence

- Generally
  $T(L) = T(L_0) + \ldots + T(L_{n-1}) + f(L)$
- Easier case: $T(L) = a\, T(L/b) + f(n)$ [MTh]
  - Ex: merge sort: $T(L) = 2\, T(L/2) + \Theta(L)$
    - Then $T(L) = \Theta(L \log L)$
  - Ex: binary search: $T(L) = T(L/2) + O(1)$
    - Then $T(L) = O(\log L)$
- Choice of threshold is important

## Long integers multiplication

- Find X*Y
  - Simple algorithms are O(m*n)
  m=log X, n=log Y
- X = 10*A+B, Y=10*C+D
- X*Y = (10*A+B)*(10*C+D) =
  = 100*A*C+10*( AD+BC)+B*D
- AD+BC = (A+B)*(C+D) - AC - BD

## Long integer multiplication

```
INT mult(INT X, INT Y) {
  if(size(X) small) return X*Y;
  split X into (A,B), split Y into (C,D)
  AC = mult(A,C); BD = mult(B,D);
  ADpBC = mult(A+B,C+D) - AC - BD;
  return (AC<<2) + (ADpBC<<1)+BD;
}
```

## Long integer multiplication

- Recurrence for X with even **size** N
  $T(N) = 3*T(N/2) + O(N)$
- MT: a=3, b=2
  $N^{\log 3}$ dominates N
- Hence for powers of two we get
  $T(N) = O(N^{\log 3})$
- Extend it to the whole range

## Long integer multiplication

- X is much shorter than Y
- M<<N
- Split Y into M-sized chunks
  - There will be N/M of those
- One smaller multiplication will take $M^{\log 3}$
- Total: $(N/M)^* M^{\log 3} = N M^{\log(3/2)}$
  - Book pp.219-223

## Finding median

- Array A[N]
- Selection problem
  - find $s^{th}$ smallest element
    - if sorted it would be in position s
  - in particular, for the median s=ceil(N/2)
- Median: as many below as above
  - Ex: [3,2,**6**,10,100,1,1000]
- Via sorting: $\Theta(N \log N)$

## Median in linear time

- Blum et al. '72
- Pivoting
  [k,l] = pivot(A[i..j], p)
    three sections [i..k] [k+1..l-1] [l..j]
- Selection call on pivoting recursively
  – If s<k<l then we are done
  – Otherwise our array is smaller, recurse

## Selection in linear time

```
float selection(A, N, s) {
   i=0; j=N-1;
   while(1) { // answer lies between i and j
         p = _median(A, i, j);
         [k,l] = pivot(A, i, j);
         if(s<=k) j=k;
         else if(s>=l) i=l;
         else return p;
   }
}
```

## Selection

- The correctness of "selection" does not depend on the choice of "_median"
  _median(A, i, j) = i
  - Results in linear time on average
    – Assume that all elements are different
    – Assume that all n! permutations are equally likely
  - Worst case is quadratic
- Better than sorting

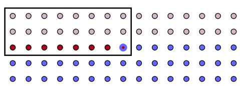
## Pseudo median

- Approximation to the median

```
float pseudo_median(A, N) {
   if(n<=5) return adhoc_median(A);
   nz = n/5;
   float Z[nz];
   for(i=0; i<nz; ++I)
         Z[i] = adhoc_median(A[5*i..5*i+4]);
   return selection(Z, ceil(nz/2));
}
```

# Pseudo median

- Approximation
  - = there is a bound on how far we are from the precise answer



  - At least 3N/10 elements are below the result
  - 3N/10 < rank(pseudo_median(A)) < 7N/10

# Runtime analysis

```
float rec_selection(A, i, j, s) { // t(N)
   p = pseudo_median(A, i, j); // t(N/5)+O(N)
   [k,l] = pivot(A, i, j); // Θ(N)
   if(s<=k)
        rec_selection(A, i, k, s); // t(7n/10)
   else if(s>=l)
        rec_selection(A, k, j, s); // t(7n/10)
   else return p;
}   /* see book pp.240-242 for more detail */
```

# Recurrence

- T(N) <= T(N/5) + T(7N/10) + O(N)
- Prove that T(N) <= K*N (will choose K below)
- Inductive step:
  - assume that T(L) <= K*L for L<N

T(N) <= K*N/5 + K*7*N/10 +O(N)
= K*N*(1/5+7/10) + O(N)
= 0.9*K*N + O(N) <= 0.9*K*N + c*N
= (0.9*K+c)*N,

choose K = 10*c, so that c=0.1*K

T(N) <= (0.9 K + 0.1 K) * N = K*N

# Matrix multiplication

- Simple algorithm $\Theta(N^3)$
- Strassen ('69)
  - Two 2x2 matrices can be multiplied using 7 multiplication instead of 8
    - Recurrence $t(n) = 7\ t(n/2) + \Theta(N^2)$
    - MT: a=7, b=2, log 7≈2.81, $n^{\log 7}$ dominates $n^2$
    - $t(n) = \Theta(N^{\log 7})$
    - Book pp. 242-243
    - Can go down to 2.376