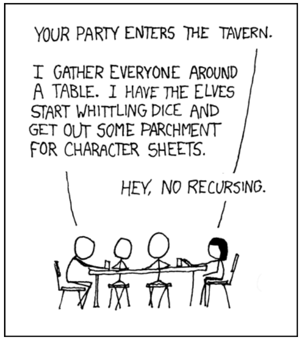
CSE 421: Algorithms

Winter 2014

Lecture 12: The Master theorem & multiplication

Reading:
Sections 5.5-5.6



sometimes two sub-problems aren't enough

- More general divide and conquer
  - You've broken the problem into a different sub-problems
  - Each has size at most n/b
  - The cost of the break-up and recombining the sub-problem solutions is $O(n^k)$
- Recurrence: $T(n) \le a \cdot T(n/b) + c \cdot n^k$

master divide and conquer recurrence

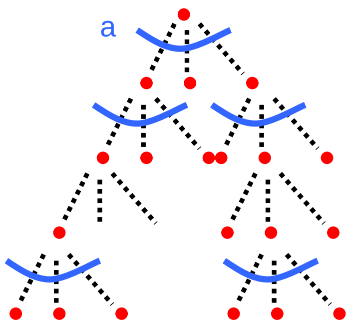
- If $T(n) \le a \cdot T(n/b) + c \cdot n^k$ for $n > b$ then
  - if $a > b^k$ then $T(n)$ is $\Theta(n^{\log_b a})$
  - if $a < b^k$ then $T(n)$ is $\Theta(n^k)$
  - if $a = b^k$ then $T(n)$ is $\Theta(n^k \log n)$

master divide and conquer recurrence

- If $T(n) \le a \cdot T(n/b) + c \cdot n^k$ for $n > b$ then
  - if $a > b^k$ then $T(n)$ is $\Theta(n^{\log_b a})$
  - if $a < b^k$ then $T(n)$ is $\Theta(n^k)$
  - if $a = b^k$ then $T(n)$ is $\Theta(n^k \log n)$
- Works even if it is $\left\lceil \frac{n}{b} \right\rceil$ instead of $\frac{n}{b}$.

proving the master recurrence

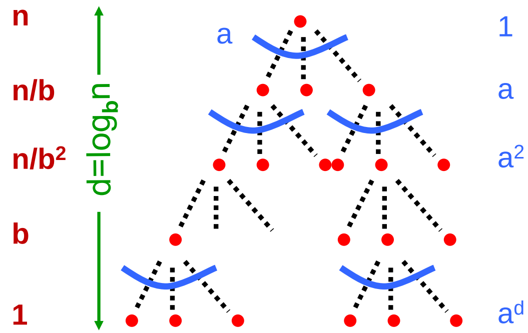
Problem size $T(n)=a \cdot T(n/b)+c \cdot n^k$ # probs



$T(1)=c$

proving the master recurrence

Problem size $T(n)=a \cdot T(n/b)+c \cdot n^k$ # probs



$T(1)=c$

geometric series

- $S \quad = t + tr + tr^2 + \ldots + tr^{n-1}$
- $r \cdot S \quad = \quad tr + tr^2 + \ldots + tr^{n-1} + tr^n$
- $(r-1)S = tr^n - t$
- so $S=t\,(r^n-1)/(r-1)$ if $r \neq 1$.

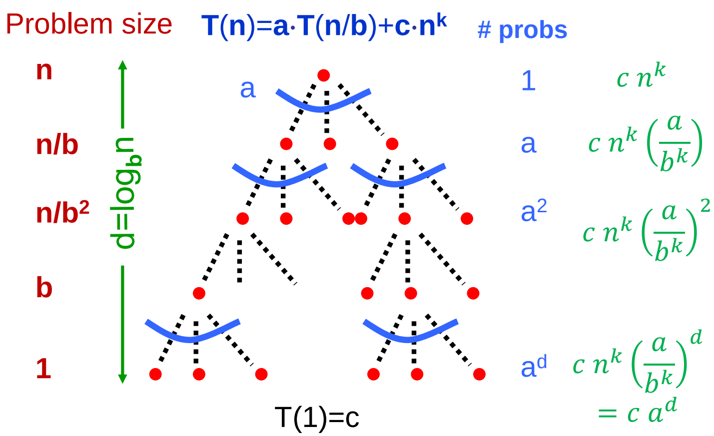
- **Simple rule**
  - If $r \neq 1$ then S is a constant times largest term in series

total cost

- Geometric series
  - ratio a/b^k
  - $d+1=\log_b n+1$ terms
  - first term cn^k, last term ca^d
- If $a/b^k=1$
  - all terms are equal $T(n)$ is $\Theta(n^k \log n)$
- If $a/b^k<1$
  - first term is largest $T(n)$ is $\Theta(n^k)$
- If $a/b^k>1$
  - last term is largest $T(n)$ is

$$\Theta\left(a^d\right)=\Theta\left(a^{\log_b n}\right)=\Theta\left(n^{\log_b a}\right)$$

proving the master recurrence



multiplying matrices

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \bullet \begin{bmatrix} b_{11} & b_{12} & b_{13} & b_{14} \\ b_{21} & b_{22} & b_{23} & b_{24} \\ b_{31} & b_{32} & b_{33} & b_{34} \\ b_{41} & b_{42} & b_{43} & b_{44} \end{bmatrix}$$

$$= \begin{bmatrix} a_{11}b_{11}+a_{12}b_{21}+a_{13}b_{31}+a_{14}b_{41} & a_{11}b_{12}+a_{12}b_{22}+a_{13}b_{32}+a_{14}b_{42} & \circ & a_{11}b_{14}+a_{12}b_{24}+a_{13}b_{34}+a_{14}b_{44} \\ a_{21}b_{11}+a_{22}b_{21}+a_{23}b_{31}+a_{24}b_{41} & a_{21}b_{12}+a_{22}b_{22}+a_{23}b_{32}+a_{24}b_{42} & \circ & a_{21}b_{14}+a_{22}b_{24}+a_{23}b_{34}+a_{24}b_{44} \\ a_{31}b_{11}+a_{32}b_{21}+a_{33}b_{31}+a_{34}b_{41} & a_{31}b_{12}+a_{32}b_{22}+a_{33}b_{32}+a_{34}b_{42} & \circ & a_{31}b_{14}+a_{32}b_{24}+a_{33}b_{34}+a_{34}b_{44} \\ a_{41}b_{11}+a_{42}b_{21}+a_{43}b_{31}+a_{44}b_{41} & a_{41}b_{12}+a_{42}b_{22}+a_{43}b_{32}+a_{44}b_{42} & \circ & a_{41}b_{14}+a_{42}b_{24}+a_{43}b_{34}+a_{44}b_{44} \end{bmatrix}$$

n^3 multiplications, $n^3 - n^2$ additions

multiplying matrices

```
for i=1 to n
   for j=1 to n
      C[i,j]←0
      for k=1 to n
         C[i,j]=C[i,j]+A[i,k]·B[k,j]
      endfor
   endfor
endfor
```

multiplying matrices

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \bullet \begin{bmatrix} b_{11} & b_{12} & b_{13} & b_{14} \\ b_{21} & b_{22} & b_{23} & b_{24} \\ b_{31} & b_{32} & b_{33} & b_{34} \\ b_{41} & b_{42} & b_{43} & b_{44} \end{bmatrix}$$

$$= \begin{bmatrix} a_{11}b_{11}+a_{12}b_{21}+a_{13}b_{31}+a_{14}b_{41} & a_{11}b_{12}+a_{12}b_{22}+a_{13}b_{32}+a_{14}b_{42} & \circ & a_{11}b_{14}+a_{12}b_{24}+a_{13}b_{34}+a_{14}b_{44} \\ a_{21}b_{11}+a_{22}b_{21}+a_{23}b_{31}+a_{24}b_{41} & a_{21}b_{12}+a_{22}b_{22}+a_{23}b_{32}+a_{24}b_{42} & \circ & a_{21}b_{14}+a_{22}b_{24}+a_{23}b_{34}+a_{24}b_{44} \\ a_{31}b_{11}+a_{32}b_{21}+a_{33}b_{31}+a_{34}b_{41} & a_{31}b_{12}+a_{32}b_{22}+a_{33}b_{32}+a_{34}b_{42} & \circ & a_{31}b_{14}+a_{32}b_{24}+a_{33}b_{34}+a_{34}b_{44} \\ a_{41}b_{11}+a_{42}b_{21}+a_{43}b_{31}+a_{44}b_{41} & a_{41}b_{12}+a_{42}b_{22}+a_{43}b_{32}+a_{44}b_{42} & \circ & a_{41}b_{14}+a_{42}b_{24}+a_{43}b_{34}+a_{44}b_{44} \end{bmatrix}$$

multiplying matrices

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \bullet \begin{bmatrix} b_{11} & b_{12} & b_{13} & b_{14} \\ b_{21} & b_{22} & b_{23} & b_{24} \\ b_{31} & b_{32} & b_{33} & b_{34} \\ b_{41} & b_{42} & b_{43} & b_{44} \end{bmatrix}$$

$$= \begin{bmatrix} a_{11}b_{11}+a_{12}b_{21}+a_{13}b_{31}+a_{14}b_{41} & a_{11}b_{12}+a_{12}b_{22}+a_{13}b_{32}+a_{14}b_{42} & \circ & a_{11}b_{14}+a_{12}b_{24}+a_{13}b_{34}+a_{14}b_{44} \\ a_{21}b_{11}+a_{22}b_{21}+a_{23}b_{31}+a_{24}b_{41} & a_{21}b_{12}+a_{22}b_{22}+a_{23}b_{32}+a_{24}b_{42} & \circ & a_{21}b_{14}+a_{22}b_{24}+a_{23}b_{34}+a_{24}b_{44} \\ a_{31}b_{11}+a_{32}b_{21}+a_{33}b_{31}+a_{34}b_{41} & a_{31}b_{12}+a_{32}b_{22}+a_{33}b_{32}+a_{34}b_{42} & \circ & a_{31}b_{14}+a_{32}b_{24}+a_{33}b_{34}+a_{34}b_{44} \\ a_{41}b_{11}+a_{42}b_{21}+a_{43}b_{31}+a_{44}b_{41} & a_{41}b_{12}+a_{42}b_{22}+a_{43}b_{32}+a_{44}b_{42} & \circ & a_{41}b_{14}+a_{42}b_{24}+a_{43}b_{34}+a_{44}b_{44} \end{bmatrix}$$

multiplying matrices

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \bullet \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$$

$$= \begin{bmatrix} A_{11}B_{11}+A_{12}B_{21} & A_{11}B_{12}+A_{12}B_{22} \\ A_{21}B_{11}+A_{22}B_{21} & A_{21}B_{12}+A_{22}B_{22} \end{bmatrix}$$

simple divide and conquer

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$$

$$= \begin{bmatrix} A_{11}B_{11}+A_{12}B_{21} & A_{11}B_{12}+A_{12}B_{22} \\ A_{21}B_{11}+A_{22}B_{21} & A_{21}B_{12}+A_{22}B_{22} \end{bmatrix}$$

$$T(n) = 8T(n/2) + 4(n/2)^2$$
$$= 8T(n/2) + n^2$$

Strassen's divide and conquer algorithm

Strassen's algorithm

- Multiply 2x2 matrices using 7 instead of 8 multiplications (and lots more than 4 additions)
- $T(n) = 7\,T(n/2) + cn^2$
  $7 > 2^2$ so $T(n)$ is $\Theta(n^{\log_2 7})$ which is $\Theta(n^{2.81})$.
- Fastest algorithms theoretically use $O(n^{2.373})$ time
  not practical but Strassen's is practical **provided calculations are exact** and we stop recursion when matrix has size somewhere between **10** and **100**

the algorithm (yuck)

$P_1 \leftarrow A_{12}(B_{11}+B_{21})$; $P_2 \leftarrow A_{21}(B_{12}+B_{22})$

$P_3 \leftarrow (A_{11} - A_{12})B_{11}$; $P_4 \leftarrow (A_{22} - A_{21})B_{22}$

$P_5 \leftarrow (A_{22} - A_{12})(B_{21} - B_{22})$

$P_6 \leftarrow (A_{11} - A_{21})(B_{12} - B_{11})$

$P_7 \leftarrow (A_{21} - A_{12})(B_{11}+B_{22})$

$C_{11} \leftarrow P_1+P_3$; $C_{12} \leftarrow P_2+P_3+P_6 - P_7$

$C_{21} \leftarrow P_1+P_4+P_5+P_7$; $C_{22} \leftarrow P_2+P_4$

multiplying faster

- If you analyze our usual grade school algorithm for multiplying numbers
  - $\Theta(n^2)$ time
  - On real machines each "digit" is, e.g., 32 bits long but still get $\Theta(n^2)$ running time with this algorithm when run on n-bit multiplication
- We can do better!
  - We'll describe the basic ideas by multiplying polynomials rather than integers
  - Advantage is we don't get confused by worrying about carries at first

notes on polynomials

These are just formal sequences of coefficients

- when we show something multiplied by x^k it just means shifted k places to the left – basically no work

Usual polynomial multiplication

$$
\begin{array}{r}
4x^2 + 2x + 2 \\
x^2 - \ 3x + 1 \\
\hline
4x^2 + 2x + 2 \\
-12x^3 - 6x^2 - 6x \quad \\
4x^4 + 2x^3 + 2x^2 \qquad\qquad \\
\hline
4x^4 - 10x^3 + 0x^2 - 4x + 2
\end{array}
$$

polynomial multiplication

- **Given:**
  - Degree n-1 polynomials P and Q

    $P = a_0 + a_1 x + a_2 x^2 + \ldots + a_{n-2}x^{n-2} + a_{n-1}x^{n-1}$

    $Q = b_0 + b_1 x + b_2 x^2 + \ldots + b_{n-2}x^{n-2} + b_{n-1}x^{n-1}$
- **Compute:**
  - Degree $2n$-2 Polynomial $P \cdot Q$
  - $P \cdot Q = a_0b_0 + (a_0b_1+a_1b_0)\,x + (a_0b_2+a_1b_1+a_2b_0)\,x^2 + \cdots + (a_{n-2}b_{n-1}+a_{n-1}b_{n-2})\,x^{2n-3} + a_{n-1}b_{n-1}\,x^{2n-2}$
- **Obvious Algorithm:**
  - Compute all a_ib_j and collect terms
  - $\Theta(n^2)$ time

naive divide and conquer

- Assume $n=2k$
  - $P = (a_0 + a_1 x + a_2 x^2 + \ldots + a_{k-2} x^{k-2} + a_{k-1} x^{k-1}) + (a_k + a_{k+1} x + \ldots + a_{n-2}x^{k-2} + a_{n-1}x^{k-1})\, x^k$
    $= P_0 + P_1 x^k$ where P_0 and P_1 are degree $k-1$ polynomials
  - Similarly $Q = Q_0 + Q_1 x^k$
- $PQ = (P_0+P_1x^k)(Q_0+Q_1x^k)$
  $= P_0Q_0 + (P_1Q_0+P_0Q_1)x^k + P_1Q_1x^{2k}$

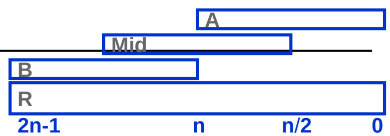
Karatsuba's algorithm

A better way to compute the terms

$$PQ = (P_0+P_1x^k)(Q_0+Q_1x^k)$$
$$= P_0Q_0 + (P_1Q_0+P_0Q_1)x^k + P_1Q_1x^{2k}$$

- Compute
  - $A \leftarrow P_0Q_0$
  - $B \leftarrow P_1Q_1$
  - $C \leftarrow (P_0+P_1)(Q_0+Q_1) = P_0Q_0+P_1Q_0+P_0Q_1+P_1Q_1$
- Then
  - $P_0Q_1+P_1Q_0 = C-A-B$
  - So $PQ=A+(C-A-B)x^k+Bx^{2k}$

Karatsuba: details




```
PolyMul(P, Q):
   // P, Q are length n =2k vectors, with P[i], Q[i] being
   // the coefficient of x^i in polynomials P, Q respectively.
   // Let Pzero be elements 0..k-1 of P; Pone be elements k..n-1
   // Qzero, Qone : similar
   If n=1 then Return(P[0]*Q[0]) else
     A ← PolyMul(Pzero, Qzero);     // result is a (2k-1)-vector
     B ← PolyMul(Pone, Qone);       // ditto
     Psum ← Pzero + Pone;           // add corresponding elements
     Qsum ← Qzero + Qone;           // ditto
     C ← polyMul(Psum, Qsum);       // another (2k-1)-vector
     Mid ← C – A – B;               // subtract correspond elements
     R ← A + Shift(Mid, n/2) +Shift(B,n)   // a (2n-1)-vector
     Return( R);
```

multiplication

- Polynomials
  - Naïve: $\Theta(n^2)$
  - Karatsuba: $\Theta(n^{1.59\ldots})$
  - Best known: $\Theta(n \log n)$
    "Fast Fourier Transform"
    FFT widely used for signal processing
- Integers
  - Similar, but some ugly details re: carries, etc. due to Schonhage-Strassen in 1971 gives $\Theta(n \log n \log\log n)$
  - Improvement in 2007 due to Furer gives $\Theta(n \log n\, 2^{\log^* n})$
  - Used in practice in symbolic manipulation systems like Maple