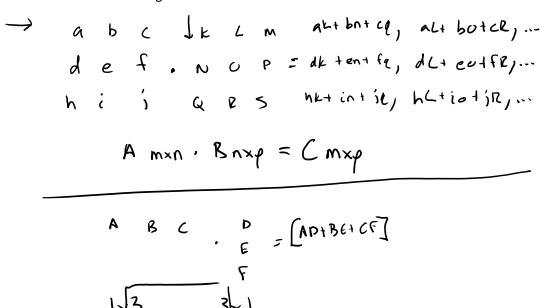
General Idea

- Goal is to identify patterns of behavior in the data or process. If
 patterns exist in a predictable way, we can often break the problem
 into a series of sub problems, which themselves can be broken down
- Divide and conquer algorithms are recursive
- Candidates for divide and conquer would be:
 - o If you notice yourself doing the same thing on subsets of the data
 - If the solution to a subsegment doesn't immediately impact adjacent subsegments
- Canonical divide and conquer algorithm: merge sort
 - Breaks data down into arrays of size 1 (which are implicitly sorted)
 and progressively merges them back together
 - From size 1, 2, 4, 8, 16....

Example #1: Matrix multiplication

• Consider the following two matrices:



Recursive, Divide and Conquer Matrix Multiplication

• Simplest matrix is of size 1

Next simplest is of size 2

$$\frac{a_{11} a_{12}}{a_{21}} = \frac{a_{11} b_{12}}{a_{21} b_{22}} = \frac{a_{11} b_{11} b_{22}}{a_{21} b_{11} b_{22}} = \frac{a_{11} b_{11} b_{22}}{a_{21} b_{11} b_{22} b_{21}} = \frac{a_{11} b_{11} b_{12} b_{22}}{a_{21} b_{11} b_{22} b_{21}} = \frac{a_{11} b_{11} b_{12} b_{22}}{a_{21} b_{11} b_{22} b_{21}} = \frac{a_{11} b_{11} b_{12}}{a_{21} b_{11} b_{22}} = \frac{a_{11} b_{11} b_{22}}{a_{21} b_{11} b_{22}} = \frac{a_{11} b_{11} b_{12}}{a_{21} b_{11} b_{22}} = \frac{a_{11} b_{11} b_{12}}{a_{21} b_{11} b_{22}} = \frac{a_{11} b_{11} b_{22}}{a_{21} b_{11} b_{22}} = \frac{a_{11} b_{11} b_{12}}{a_{21} b_{21} b_{22}} = \frac{a_{11} b_{11} b_{12}}{a_{21} b_{21} b_{22}} = \frac{a_{11} b_{12} b_{22}}{a_{21} b_{11} b_{22}} = \frac{a_{11} b_{11} b_{22}}{a_{21} b_{11} b_{22}} = \frac{a_{11} b_{11} b_{22}}{a_{21} b_{11} b_{22}} = \frac{a_{11} b_{11} b_{22}}{a_{21} b_{11} b_{22}} = \frac{a_{11} b_{12} b_{22}}{a_{21} b_{11} b_{22}} = \frac{a_{11} b_{11} b_{12}}{a_{21} b_{21} b_{22}} = \frac{a_{11} b_{11} b_{12}}{a_{21} b_{21} b_{22}} = \frac{a_{11} b_{11} b_{22}}{a_{21} b_{21} b_{22}} = \frac{a_{11} b_{11} b_{22}}{a_{$$

only warks on size

- recurse (a_{s,eh}; b_{s,eh}) + recurse (a_{y,e}) b_{e 15})
- Each recursive step makes 8 recursive calls
- Each recursive call handles n/2 less data
- At each recursive call, we do N^2 each time

$$a_{11} \cdot a_{12} \cdot a_{13} \cdot a_{14}$$
 $a_{11} \cdot a_{12} \cdot a_{13} \cdot a_{24}$
 $a_{21} \cdot a_{22} \cdot a_{23} \cdot a_{24}$
 $a_{31} \cdot a_{32} \cdot a_{33} \cdot a_{34}$
 $a_{41} \cdot a_{42} \cdot a_{43} \cdot a_{44}$
 $a_{41} \cdot a_{42} \cdot a_{43} \cdot a_{44}$
 $a_{42} \cdot a_{43} \cdot a_{44}$
 $a_{43} \cdot a_{44} \cdot a_{45} \cdot a_{44}$
 $a_{44} \cdot a_{44} \cdot a_{45} \cdot a_{44}$
 $a_{45} \cdot a_{45} \cdot a_{45}$
 $a_{46} \cdot a_{47} \cdot a_{47} \cdot a_{47}$

• Strassen's Method reduces the number of recursive calls from 8 to 7

$$T(n) = 7T(2) + n^{2}$$

$$\Theta(n^{\omega_{2}7}) = O(n^{2.81})$$

Maximum Subsequence Problem

• Given a series of positive and negative integers, find the subsequence that yields the maximum sum

• Given a series of positive and negative integers, find the subsequence that yields

