Slide 1

Administrivia

- Reminder: Homework 1 due Friday.
- About e-mail to me: You may notice that the mail you get from me comes from an address other than my official TMail address (it has a `@cs` in the middle). Why that is — a little complicated, a matter of tools preference and habit, not that interesting. Anyway use whichever you find more convenient, but if you send to both, I get two copies of your message, which I'd rather not.

Slide 2

Recursive Matrix Multiplication, Take 2 — Strassen's Algorithm

- It probably seems intuitively obvious that execution time for matrix multiplication would be at best $\Theta(n^3)$, and this was widely believed for a long time, but then a mathematician named Strassen came up with an algorithm that runs in $\Theta(n^{\log 7})$ time, which isn't as dramatic an improvement as mergesort over insertion sort but is an improvement!
- It does this by reducing the number of recursive steps from eight to seven . . .
- First, a terminology convention: We'll use "addition" to mean addition or subtraction, since they're basically the same operation (and if you've taken CSCI 2321 you more or less know why!).

Slide 3

Strassen's Algorithm, Continued

- Strassen's algorithm is based on applying an observation from basic algebra:
- Computing $x^2 - y^2$ involves two multiplications and one addition. But you may remember that

$$x^2 - y^2 = (x + y)(x - y)$$

and the expression on the right involves only one multiplication and two additions, which is likely faster given that in general multiplication tends to be slower than addition. If we apply this to matrix multiplication, the difference is likely more pronounced. So . . .

Slide 4

Strassen's Algorithm, Continued

- The base case is the same as for the simple recursive algorithm — when $n = 1$.
- For the recursive case, we start by partitioning the matrices as for the simple algorithm.
- We then compute 10 matrices that are sums and differences of two of these submatrices, and seven products of these sums (using recursion to compute the products).
- We then use these product matrices to update the four submatrices of C.
- In more detail . . .

Slide 5

Strassen's Algorithm, Continued

- The sum / difference matrices:

$$
\begin{aligned}
S_1 &= B_{12} - B_{22} & S_6 &= B_{11} + B_{22} \\
S_2 &= A_{11} + A_{12} & S_7 &= A_{12} - A_{22} \\
S_3 &= A_{21} + A_{22} & S_8 &= B_{21} + B_{22} \\
S_4 &= B_{21} - B_{11} & S_9 &= A_{11} - A_{21} \\
S_5 &= A_{11} + A_{22} & S_{10} &= B_{11} + B_{12}
\end{aligned}
$$

Slide 6

Strassen's Algorithm

- The product matrices (note that each must first be initialized to 0):

$$
\begin{aligned}
P_1 &= A_{11} \cdot S_1 \\
P_2 &= S_2 \cdot B_{22} \\
P_3 &= S_3 \cdot B_{11} \\
P_4 &= A_{22} \cdot S_4 \\
P_5 &= S_5 \cdot S_6 \\
P_6 &= S_7 \cdot S_8 \\
P_7 &= S_9 \cdot S_{10}
\end{aligned}
$$

Strassen's Algorithm

Slide 7

- Finally, use these matrices to compute the submatrices of C (which has first been initialized to 0):

$$
\begin{aligned}
C_{11} &= C_{11} + P_5 + P_4 - P_2 + P_6 \\
C_{12} &= C_{12} + P_1 + P_2 \\
C_{21} &= C_{21} + P_3 + P_4 \\
C_{22} &= C_{22} + P_5 + P_1 - P_3 + P_7
\end{aligned}
$$

Strassen's Algorithm, Correctness

Slide 8

- It's tedious but presumably not too difficult to confirm through some rather cumbersome algebra that this gives the same result for the submatrices of C as a straightforward calculation based on the definition. The textbook gives all the details. (A conscientious person would follow them through carefully. By that definition, alas . . .)
- (I do wonder how the algorithm's inventor came up with it!)

Strassen's Algorithm, Analysis

Slide 9

- Partitioning the matrices takes constant time, just like the simple recursive algorithm, if we use index calculations.
- Creating matrices S_n and zeroing the submatrices of C takes time $\Theta(n^2)$.
- Computing matrices P_n takes time $7T(n/2)$ time, where $T(n)$ as usual denotes execution time for using the algorithm on matrices of size n by n.
- Updating the submatrices of C involves additions only, so it takes time $\Theta(n^2)$.
- This gives a recurrence relation of

$$T(n) = 7T(n/2) + \Theta(n^2)$$

Strassen's Algorithm, Analysis Continued

Slide 10

- The recurrence relation

$$T(n) = 7T(n/2) + \Theta(n^2)$$

  fits a form to which the master theorem applies, and the theorem says execution time is $\Theta(n^{\log 7})$
- Recall that the simple approach gives an execution time of $\Theta(n^3)$. Since $3 = \log 8 > \log 7$, this method is indeed faster, at least a little!

Slide 11

Minute Essay

- Questions?