15-451/651 Algorithms

15-451/651
9/3/20

Instructors: Gary Miller
Klaus Sutner

TAs:

Grading & Requirements

- 2 Lectures & 1 Recitation per week.
- 1 Quiz per week.
- HW every other week.
- 2 Midterm & Final.

- Grading:
- 4- Written-HW 20%
- 3- Oral-HW 15%
- Quizzes 12%
- 2- Midterms 30%
- Final 23%

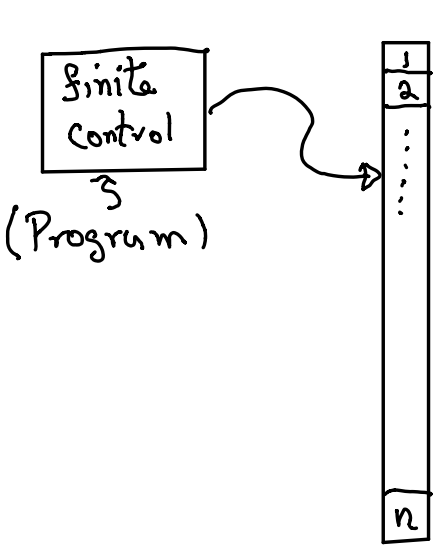
Discussion Piazza
Webpage: Find from ~glmiller
Grades A to R
Audit I Pass/Fail

Course Goals:

1) Understand many known:
   a) Algorithms.
   b) Design Techniques
2) Analyze algorithm efficiency
3) Algorithm correctness
4) Communicate about code.
5) Know key words
6) Design your own algs.
7) Consider examples from many areas of CS.
   a) System
   b) Graphics
   c) Theory
   d) ML
   e) Programming Languages.

Machine Model Mostly Used

RAM ≡ Random Access Machine



unit time operations

1) Memory read/write
2) Arithmetic ops
   $+, -, \times, \div$
   ($\log n$ bit size)

not considered

Caching Models

1) Memory Hierarchy
2) Pipeling

Parallel Models

PRAM

Circuits

Asymptotic Complexity

$f: \mathbb{R}^+ \to \mathbb{R}$ $\quad g: \mathbb{R}^+ \to \mathbb{R}$

Def: $f \in O(g)$ if $\quad (f = O(g))$

$\exists n_0, c \geq 0 \quad \forall n \geq n_0 \quad f(n) \leq c\, g(n)$

i.e $O(g(n)) = \{ f \mid f \in O(g) \}$

Def: $f \in o(g)$ if

$\forall c > 0 \quad \exists n_0 \geq 0 \quad \forall n \geq n_0 \quad f(n) \leq c\, g(n)$

---

1) $f \in \Omega(g)$ if $g \in O(f)$

2) $f \in \Omega(g)$ if

$\exists c \geq 0 \quad \forall n_0 \geq 0 \quad \exists n_1 \geq n_0 \quad f(n_1) \geq c\, g(n_1)$

(infinitely often)

(2 possible def)

An Example

Claim: $2n^2+n+1 \in O(n^2)$

Guess that $C=3$ works

to show $2n^2+n+1 \leq 3n^2$ for $n \geq n_0$

Or $n^2-n-1 \geq 0$ for $n \geq n_0 = 2$

(check by taking derivative)

Thus: $\forall n \geq 2 \; (2n^2+1+1) \leq 3n^2$

L'Hopitals Rule

$$\lim_{n\to\infty} \frac{2n^3+n+1}{n^2} = \lim_{n\to\infty} \frac{4n+1}{2n}$$

$$= \lim_{n\to\infty} \frac{4}{2} = 2$$

Thus: $C = 2+\varepsilon$ works

Example 2

Claim: $3n \in o(n^2)$

We need n_0 to be a fcn of c.

Solve $3n = cn^2$

$3 = cn$

$n = 3/c$

Set $n_0 = 3/c$

to show: $3n \leq cn^2$ for $n \geq 3/c$

iff $3 \leq cn^2$

iff $3/c \leq n$

L'Hopital

$\lim_{n \to \infty} \frac{3n}{n^2} = 0$ thus $3n \in o(n^2)$

Finding a Triangle in a Graph

Input: Graph $G=(V,E)$

Question: $\exists\, a,b,c \in V$ s.t. $(a,b),(b,c),(a,c) \in E$
(triangle) (3-cycle)

An $O(n^3)$ algorithm

1) $\forall$ distinct $a,b,c \in V$ check if $\{a,b,c\}$ is triangle.

Note 1) Number triples $= O(n^3)$

2) Cost to check if a $\triangle$ is $O(1)$.

3) $O(n^3)$ algorithm.

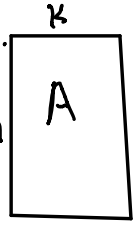
Can we do better?

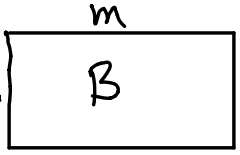
We view G as an adj matrix A and use matrix multiplication.

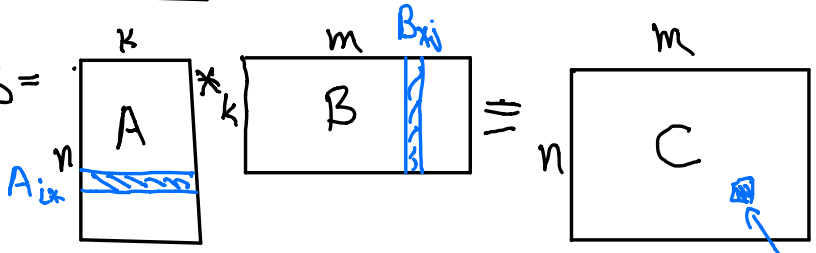
Matrix Multiplication

Dot Product: $a = (a_1, \dots, a_n)$ & $b = (b_1, \dots, b_n)$ $\quad a_i, b_i \in \mathbb{R}$

$$a \cdot b \equiv \sum_{i=1}^{n} a_i \cdot b_i$$

A^{nk} = 

B^{km} = 

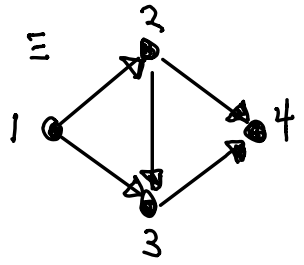
$A \cdot B =$ 

where $C_{ij} = \sum_{t=1}^{k} A_{ik} \cdot B_{kj} \equiv A_{i*} \cdot B_{*j}$

Important facts

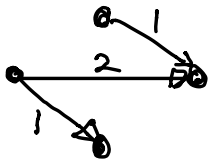
1) $(A \cdot B)C = A(B \cdot C)$
2) $A(B + C) = AB + AC$
3) $A \cdot B \neq BC$ (in general)
4) $\lambda A \equiv \begin{pmatrix} \lambda & & \\ & \ddots & \\ & & \lambda \end{pmatrix} A$
5) $C = AB$ then $C = \sum_{j=1}^{k} A_{*j} \cdot B_{j*}$ (outer products)

Matrix Multiplication on Adj Matrices

$G \equiv$ 

$$A_G = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

$$A^2 = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

$G^2 =$ 

Thus $(A^2)_{ij} \equiv$ # of 2-edge paths from i to j.

Δ-Alg 1) Compute A^2
2) return all (i,j) s.t. $A^2_{ij} \neq 0$ & $A_{ji} \neq 0$

Note: Step 2) is $O(n^2)$

Question: How fast can we compute A^2?

Question: Can we count number of Δ's in G?

What about $\frac{1}{3} \sum_{A_{ji} \neq 0} (A^2)_{ij}$?

Matrix Multiplication Alg

(Naive) If A, B are $n \times n$ matrices

For $1 \le i, j \le n$

return $c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj}$

We get n^3 multiplications

$(n-1)n^2$ additions

$O(n^3)$ operations

Recursive Alg

Assume $n = 2^k$

$M(A, B)$

1) if A is 1×1 then return $a_{11} \cdot b_{11}$

2) Write $A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}, B = \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix}$

A_{ij} & B_{ij} are $n/2 \times n/2$

3) Compute for $1 \le i, j \le 2$

$$C_{ij} = M(A_{i1}, B_{1j}) + M(A_{i2}, B_{2j})$$

(note) $(A+B)_{ij} = a_{ij} + b_{ij}$

4) Return $\begin{pmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{pmatrix}$

Correctness (induction on n)

1) $n=1$ (done)

By induction assume $M(A,B) = A \cdot B$ $(n < n_0)$

By property of mult of matrices

$$C_{ij} = A_{i1} \cdot B_{1j} + A_{i2} \cdot B_{2j}$$

Thus $C_{ij} = M(A_{i1}, B_{ij}) + M(A_{i2}, B_{2j})$

Timing: Let $T(n)$ = number of ops for $n \times n$

$$T(n) \leq 8T(n/2) + cn^2 \quad \& \quad T(1) = 1$$

We will show $T(n) = O(n^3)$

Consider recurrence

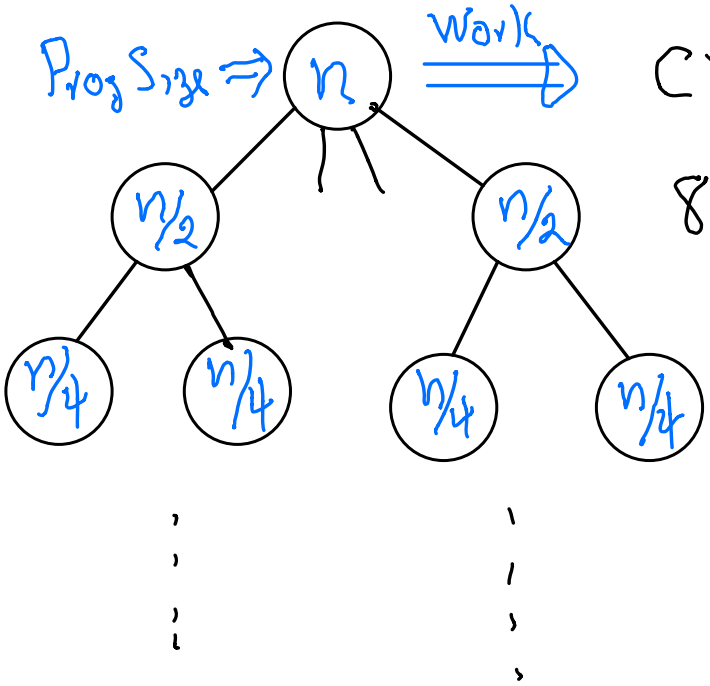
$$T(n) \leq 7T(n/2) + cn^2 \quad \& \quad T(1) = 1$$

Claim: $T(n) = O(n^{\log_2 7}) = O(n^{2.81\ldots})$

Solving Recurrences

Methods: 1) Use Formula
2) Induction on n.
3) Tree of recursive calls

3)



cn^2

$8c(n/2)^2 = 2cn^2$

$(2^3)^2 c(n/2^2)^2$

$= 2^2 cn^2$

1

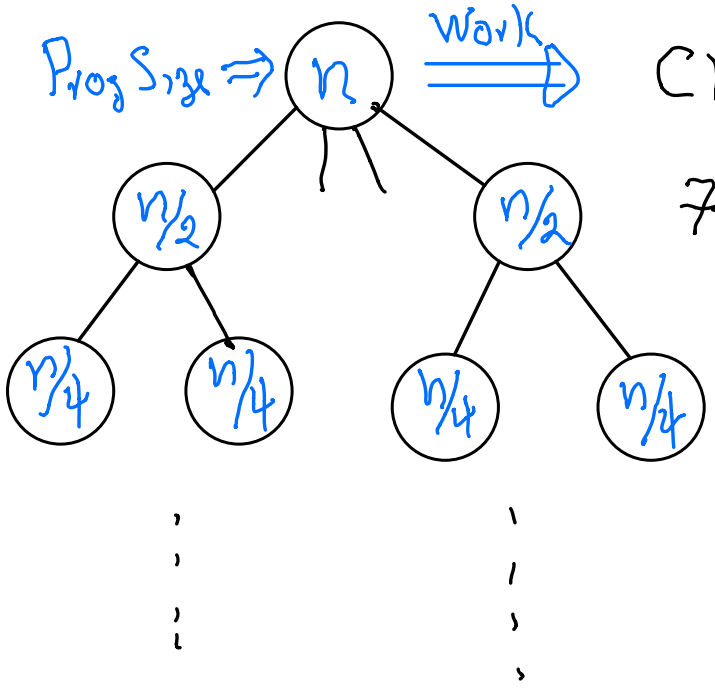
1

$2^{\log n} cn^2$

$cn^3 = O(n^3)$

Consider $T(n) = 7T(n/2) + cn^2$

3)



cn^2

$7c(n/2)^2 = (7/2)cn^2$

$7^2 c(n/4)^2$

$= (7/4)^2 cn^2$

(1) (1) $2^{\log n} cn^2$

$$\left(\frac{7}{4}\right)^{\log n} cn^2 = \frac{n^{\log 7}}{n^{\log 4}} cn^2$$

$$= cn^{\log 7}$$

Total $O(n^{\log 7})$

Strassen's Matrix Mult.

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix}\begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} S_1+S_2-S_4+S_6 & S_4+S_5 \\ S_6+S_7 & S_2-S_3+S_5-S_7 \end{bmatrix}$$

$$S_1 = (B-D)(G+H)$$
$$S_2 = (A+D)(E+H)$$
$$S_3 = (A-C)(E+F)$$
$$S_4 = (A+B)\cdot H$$
$$S_5 = A\cdot(F-H)$$
$$S_6 = D\cdot(G-E)$$
$$S_7 = (C+D)\cdot E$$

7 recursive calls!

Correctness (4 cases)

Case $C_{21} = C \cdot E + D \cdot G \stackrel{?}{=} S_6 + S_7$

$$S_6 + S_7 = D(G - E) + (C + D)E$$
$$= DG - DE + CE + DE$$
$$= CE + DG$$

Thm Matrix Multi of $n \times n$ matrices is $O(n^{2.81\ldots})$

Thm Counting Δ is $O(n^{2.81\ldots})$ time.

6.5 Another version of Strassen's algorithm uses the following identities to help compute the product of two 2×2 matrices.

$$
\begin{array}{lll}
s_1 = a_{21} + a_{22} & m_1 = s_2 s_6 & t_1 = m_1 + m_2 \\
s_2 = s_1 - a_{11} & m_2 = a_{11} b_{11} & t_2 = t_1 + m_4 \\
s_3 = a_{11} - a_{21} & m_3 = a_{12} b_{21} & \\
s_4 = a_{12} - s_2 & m_4 = s_3 s_7 & \\
s_5 = b_{12} - b_{11} & m_5 = s_1 s_5 & \\
s_6 = b_{22} - s_5 & m_6 = s_4 b_{22} & \\
s_7 = b_{22} - b_{12} & m_7 = a_{22} s_8 & \\
s_8 = s_6 - b_{21} & &
\end{array}
$$

The elements of the product matrix are:

$$
\begin{aligned}
c_{11} &= m_2 + m_3, \\
c_{12} &= t_1 + m_5 + m_6, \\
c_{21} &= t_2 - m_7, \\
c_{22} &= t_2 + m_5.
\end{aligned}
$$

Show that these elements compute Eq. (6.1). Note that only 7 multiplications and 15 additions have been used.

What is a Space Efficient Strassen

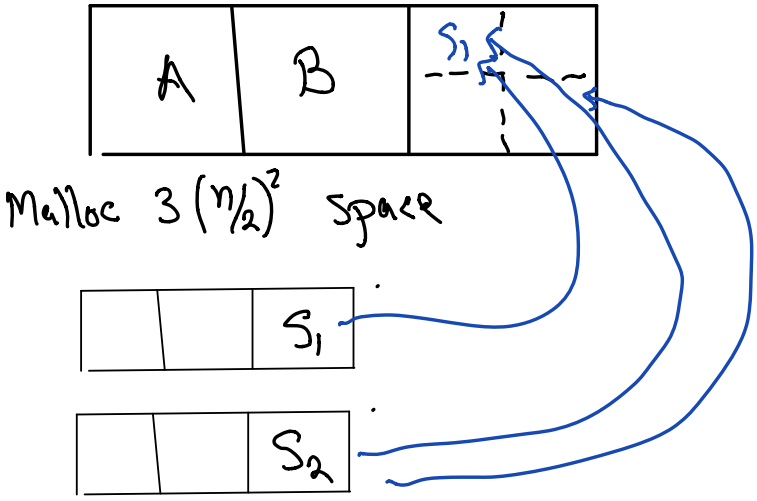
Alg 1) Add in place

2) Malloc $3n^2$ space per call.

3) Do final additions in output space of parent.

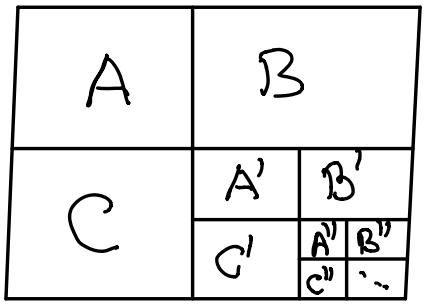
---

eg Malloc $3n^2$ space



---

eg



As a recurrance

Let $W(n)$ be space used

$$W(n) = 3n^2 + W(n/2)$$
$$= 3n^2 + 3(n/2)^3 + 3(n/4)^2$$
$$= 3n^2(1 + 1/4 + 1/16 + \cdots)$$

---

note $1 + x + x^2 + \cdots = \frac{1}{1-x} \quad x < 1$

$$W(n) = 3n^2\left(\frac{1}{1-1/4}\right) = 3n^2(4/3) = 4n^2$$