CS 809
Lecture Notes

Eric Bach
University of Wisconsin
Computer Sciences Dept.

Foreword

These are lecture notes from a graduate course on mathematical techniques for algorithm analysis, which I have taught in Wisconsin's computer science department since 1990.

Initially the course dealt with methods for exact analysis of algorithms, and lower bounds in concrete models of computation. In later editions, the material on lower bounds was taken out to give more room for recent probabilistic ideas, such as martingales and Markov chain Monte Carlo algorithms.

Comments and corrections are most welcome.

Table of Contents

38. Second moment method
39. Erdos probabilistic method

IV. Markov Chains and Branching Processes
40. Markov chains, Monte Carlo
41. MCMC for graph coloring
42. Graph coloring MCMC, continued
43. Branching processes
44. Tree search with random costs
45. Branching random walks

V. Lower Bounds
46. Intro to lower bounds
47. Entropy bounds for sorting, merging, etc.
48. Searching in partial orders
49. Poset searching (continued)
50. Geometric ideas for lower bounds
51. Geometric lower bound ideas (continued)
52. Algebraic decision trees
53. Adversary strategies
54. Algebraic complexity theory
55. Lower bounds from linear algebra
56. Quadratic and bilinear forms, Tensor rank
57. Lower bounds from algebraic geometry

Lecture 1

Topics du jour: Course introduction, record value problem.

What's this about?

See course flyer and syllabus.

The best way to introduce this material is to actually do some of it. So we will analyze a simple algorithm.

Record updating

Given a sequence of numerical samples, we want to keep track of the largest one seen so far.

A (straightforward) algorithm for this:

Let the samples be $X_1, \ldots, X_n$.

Set $M := -\infty$.

For $i := 1$ to n, replace M by X_i if $X_i > M$.

An evaluation of the instruction $M := X_i$ is called an *update*.

How many updates?

Best case: $X_1 \geq X_2, \ldots, X_n$, which has 1 update.

Worst case: $X_1 < X_2 < \cdots < X_n$, which forces n updates.

We will show that the average is much better than the worst case: it is asymptotic to $\log n$.

Thus, there is an exponential gap between the expected value and the worst case.

A probabilistic model

If we want to talk about the "average" behavior of an algorithm, we have to decide on some probability distribution for the input.

The right model for this problem is that the X_i are i.i.d. uniform samples from the interval $(0, 1)$.

Why did we choose the uniform distribution?

Any other distribution with a density would lead to the same results. Suppose that X is an absolutely continuous random variable (this means that X is given by a density, so that $F(x) := \Pr[X \leq x] = \int_{-\infty}^{x} f(t)dt$). Then, the random variable $Y = F(U)$ has a uniform distribution on $(0, 1)$.

This is one of the key facts about order statistics, and leads to distribution-free results.

If $X_1, \ldots, X_n$ is a sequence of numbers, the sorted values

$$X_{(1)} \le X_{(2)} \le \cdots \le X_{(n)}$$

are called the *order statistics*.

With probability 1, these are distinct. Therefore, there is a unique permutation σ for which

$$X_{(i)} = X_{\sigma(i)}, \qquad i = 1, \ldots, n.$$

Each σ is equally likely.

You can check this for small n. For example, if X and Y are two independent uniform samples, it is equally likely that $X < Y$ or $X > Y$.

For this reason, we may as well take our data to be a random ordering of the integers $1, \ldots, n$.

Records in random permutations

Let $X_1, \ldots, X_n$ be a permutation of $1, \ldots, n$. A *record* is a value X_i that is larger than every X_j for $j < i$.

How many records does a random permutation of $1, \ldots, n$ have?

To answer this, we give a method to construct a random permutation.

Start with L empty.

For $i := n$ down to 1, insert i into a randomly chosen "gap" in the list (of size $n - i$) constructed so far.

Example: for $n = 5$ we might produce

5
4 5
4 3 5
2 4 3 5
2 4 3 1 5

Note that the records are exactly the elements inserted at the left of the lists. So this permutation has the records 2,4,5.

Define random variables Y_i, by

$$Y_i = \begin{cases} 1, & \text{if } i \text{ was inserted at the left;} \\ 0, & \text{otherwise.} \end{cases}$$

This is a standard trick in probability theory, useful whenever you have a random variable that counts something. The Y_i's are called *indicator variables*.

Then the number of records is $\sum_{i=1}^{n} Y_i$.

We have

$$\begin{aligned} E[Y_n] &= 1; \\ E[Y_{n-1}] &= 1/2; \\ E[Y_{n-2}] &= 1/3; \\ &\cdots \\ E[Y_{n-i}] &= 1/(i+1). \end{aligned}$$

So

$$\begin{aligned} E[\text{ \# of records }] &= E[\sum Y_i] \\ &= \sum E[Y_i] \\ &= \sum_{i=0}^{n-1} 1/(i+1) \\ &= \sum_{i=1}^{n} 1/i. \end{aligned}$$

This is a standard sum, called the n-th *harmonic number*. We denote it by H_n.

Facts about harmonic numbers.

By comparing the sum and the integral you can see

$$\log n \leq H_n \leq \log n + 1.$$

It can be shown (using Euler-Maclaurin summation) that

$$H_n = \log n + \gamma + O(1/n),$$

where $\gamma = 0.57721...$ is Euler's constant.

What's the variance? The Y_i's are independent, so

$$\begin{aligned} \sigma^2[\text{ \# of records }] &= \sum \sigma^2[Y_i] \\ &= \sum E[Y_i^2] - E[Y_i]^2 \\ &= \sum E[Y_i] - E[Y_i]^2 \\ &= \sum_{i=1}^{n} 1/i - 1/i^2. \end{aligned}$$

It can be shown that

$$\sum_{i \geq 1} 1/i^2 = \frac{\pi^2}{6}.$$

Therefore,

$$\sigma^2[\ \#\text{ of records }] = \log n + (\gamma - \pi^2/6) + O(1/n).$$

You can see that we are getting answers that are very precise.

Using the central limit theorem (which says that the sum of a large number of well-behaved independent random variables is asymptotically normally distributed), we can show that

$$\Pr[\alpha\sqrt{\log n} < \#\text{ of records} - H_n < \beta\sqrt{\log n}] \sim \frac{1}{\sqrt{2\pi}} \int_\alpha^\beta e^{-t^2/2} dt.$$

We can summarize by saying that the expected number and standard deviation for the number of updates is asymptotic to $\log n$.

Some actual data

The *Sporting News* handbook gives the American League home run champions for the years 1901-1960.

This is a good period to look at, because the rules of baseball were relatively constant. (There were 8 clubs in the league, with 154 games in a season.)

The "record" values in this series come from years in which the champion hit more home runs than anyone previously. They are:

1901 – Nap Lajoie – 14 home runs
1902 – Socks Seybold – 16 home runs
1919 – Babe Ruth – 29 home runs
1920 – Babe Ruth – 54 home runs
1921 – Babe Ruth – 59 home runs
1927 – Babe Ruth – 60 home runs

This gives 6 record values in a 60 year period.

What do we expect?

The mean $H_{60} \doteq 4.68$, with $\sigma \doteq 1.74$.

A value of 6 is therefore within reason.

Notes

The decomposition of the record count into independent indicators is due to Alfred Rényi, Ann. Sci. Univ. Clermont-Ferrand 2, v. 8, sér. Math., no. 2, 1962, pp. 7-13.

Some nice applications of this: average performance of Prim minimum spanning tree (Martel, IPL 2002); average performance of Dijkstra's shortest path algorithm (Noshita, J. Algorithms 6, 1986, 400-408).

The discrete record value problem for geometric samples can be used to analyse skip lists (Pugh, CACM around 1990).

Lecture 2

Topics du jour: Binomial coefficients, binomial distribution.

References: PB 3.2, GK 1.2, Feller 1 IV.2.

Binomial coefficients

We define

$$\binom{n}{k}$$

(read "n choose k") to be the number of ways we can choose a subset of size k from a universe of size n.

By considering the possible choice sequences, we get

$$\binom{n}{k} = \frac{n(n-1)\cdots(n-k+1)}{k!} = \frac{n!}{(n-k)!k!}$$

The k-th *falling power* of n is

$$n^{\underline{k}} = n(n-1)\cdots(n-k+1).$$

Using this notation, $\binom{n}{k} = n^{\underline{k}}/k!$.

From the definition it follows that

$$\binom{n}{k} = 0$$

for $k < 0$ and $k > n$.

It is useful to know that

$$\binom{n}{k} = \frac{n}{k}\binom{n-1}{k-1}$$

(when $k \neq 0$.)

When the upper index is not integral, we define the binomial coefficient by

$$\binom{x}{k} = \begin{cases} x^{\underline{k}}/k!, & \text{if } k \geq 0; \\ 0, & \text{if } k < 0. \end{cases}$$

(Here k is an integer.)

Binomial theorem:

$$(1+x)^n = \sum_{i \geq 0} \binom{n}{i} x^i.$$

Valid for any x when n is a non-negative integer.

Otherwise, take the left side to be the real positive value of $(1+x)^n$. The right hand side converges to this for $|x| < 1$.

By plugging in various values for x, we can evaluate binomial sums:

$$\sum_{i=0}^{n} \binom{n}{i} = 2^n. \qquad \text{Take } x = 1.$$

$$\sum_{i=0}^{n} (-)^i \binom{n}{i} = 0. \qquad \text{Take } x = -1.$$

Recurrence relation:

$$\binom{n}{i} = \binom{n-1}{i-1} + \binom{n-1}{i}$$

Combinatorial interpretation: Choose i elements from $\{1, \ldots, n\}$. If we include n in our set, we can pick the other elements in $\binom{n-1}{i-1}$ ways. If we do not, the others come from a set of size $n-1$, in $\binom{n-1}{i}$ ways.

Pascal's triangle (n-th row gives $\binom{n}{i}$, $i = 0, \ldots, n$):

					1							
				1		1						
			1		2		1					
		1		3		3		1				
	1		4		6		4		1			
1		5		10		10		5		1		
1		6		15		20		15		6		1

Note that:

$\binom{n}{i} = \binom{n}{n-i}$ (symmetry).

Any entry is the sum of the two diagonally above it (recurrence relation).

The sum of the first k diagonal ($\searrow$) entries appears immediately below and to the left of the last one. In symbols, we have

$$\sum_{k=0}^{m} \binom{n+k}{k} = \binom{n+m+1}{m}.$$

From the other diagonal ($\swarrow$) we get

$$\sum_{m=0}^{n} \binom{m}{k} = \binom{m+1}{k+1}.$$

This will be proved later.

Binomial distribution.

Let X_i, $i = 1, \ldots, n$ be i.i.d. random variables, equal to 1 with probability p and 0 with probability $q = 1 - p$.

We call the X_i's *Bernoulli trials*.

By definition,

$$X = \sum_{i=1}^{n} X_i$$

has the *Binomial(n,p)* distribution.

Explicitly

$$\Pr[X = i] = \binom{n}{i} p^i q^{n-i}.$$

Why? There are $\binom{n}{i}$ places to put the 1's. Each of these patterns occurs with the (same) probability $p^i q^{n-i}$.

Mean and variance:

$$E[X] = \sum_i E[X_i] = np.$$

$$\begin{aligned}
\sigma^2[X] = & \sum_i \sigma^2[X_i] \text{ (by independence)} \\
= & \sum_i E[X_i^2] - E[X_i]^2 \\
= & \sum_i E[X_i] - \sum_i E[X_i]^2 \\
= & \ n(p - p^2) = npq.
\end{aligned}$$

Binomial sums.

When confronted with a sum to be evaluated, try to relate it to the moments of a known probability distribution. You will be surprised at how often this idea works. Some examples:

1. $$\sum_{i=1}^{n} i \binom{n}{i} = 2^n \sum_{i=1}^{n} i \binom{n}{i} (1/2)^n = 2^n E[\text{Binomial}(n, 1/2)] = n 2^{n-1}.$$

2. $$\sum_{i=1}^{n} i^2 \binom{n}{i} = 2^n E[X^2],$$

where X has the Binomial(n,p) distribution. So this sum equals

$$2^n\left(\sigma^2[X]+E[X]^2\right)=2^n(n/4+n^2/4)=(n^2+n)2^{n-2}.$$

Higher moments of binomials

These are easiest to remember with the formula

$$E\left[\binom{X}{k}\right]=p^k\binom{n}{k}.$$

To prove this, observe that

$$\begin{aligned}
E[X^{\underline{k}}] &= \sum_i i^{\underline{k}}\binom{n}{i}p^iq^{n-i}\\
&= \sum_i i^{\underline{k}}\,\frac{n^{\underline{k}}}{i^{\underline{k}}}\binom{n-k}{i-k}p^iq^{n-i}\\
&= n^{\underline{k}}p^k\sum_i\binom{n-k}{i-k}p^{i-k}q^{n-i}\\
&= n^{\underline{k}}p^k\sum_j\binom{n-k}{j}p^jq^{n-k-j}\\
&= n^{\underline{k}}p^k.
\end{aligned}$$

(I learned this from Huang Jiansheng.)

There is a nice "indicator variable" proof of this one, too. Imagine that we have n bits, each of which is turned on with probability p. Let σ be a k-subset of the bit positions, and let Y_σ be the indicator that is 1 if all positions in σ are on, 0 otherwise. Then

$$\binom{X}{k}=\sum_\sigma Y_\sigma.$$

Take expectations of both sides.

More binomial sums

We can now sum anything of the form $\sum f(i)\binom{n}{i}$ when f is a polynomial.

We illustrate with $\sum i^3\binom{n}{i}$.

We know that

$$i^3=i^{\underline{3}}+Ai^{\underline{2}}+Bi.$$

Taking $i=1$ we find $B=1$.

Taking $i=2$ we find $8=2A+2$, so $A=3$.

Therefore

$$\begin{aligned}\sum i^3 \binom{n}{i} &= 2^n E[X^3] \\ &= 2^n E[X^{\underline{3}} + 3X^{\underline{2}} + X] \\ &2^n \left(\frac{n^{\underline{3}}}{8} + 3\frac{n^{\underline{4}}}{4} + n \right) \\ &2^{n-3} \left(n^3 + 2n^2 \right).\end{aligned}$$

Probability generating functions

This is another approach to moment computation.

Let X be a random variable taking the values $0, 1, 2, \ldots$.

The probability generating function (PGF) of X is

$$F(t) = \sum_{m \geq 0} \Pr[X = m] t^m.$$

For the binomial distribution, we have

$$F(t) = \sum_m \binom{n}{m} p^m q^{n-m} t^m = (pt + q)^n.$$

$F(1+t)$ contains information about the moments of X, because of the following:

$$\begin{aligned}F(1+t) = & \sum_{m \geq 0} \Pr[X = m](1+t)^m \\ = & \sum_{m \geq 0} \Pr[X = m] \sum_{j \geq 0} \binom{m}{j} t^j \\ = & \sum_{j \geq 0} \left(\sum_{m \geq 0} \binom{m}{j} \Pr[X = m] \right) t^j \\ = & \sum_{j \geq 0} E\left[\binom{X}{j} \right] t^j.\end{aligned}$$

In other words, we have

$$\begin{aligned}F(1) = & \ 1, \\ F'(1) = & \ E[X], \\ F''(1) = & \ E[X(X-1)],\end{aligned}$$

and in general

$$F^{(k)}(1) = E[X^{\underline{k}}].$$

Some people refer to values like $E[X(X-1)(X-2)]$ as "factorial moments." So $F(1+t)$ is a factorial moment-generating function. The ordinary moment-generating function (Laplace transform) of X is $E[e^{tX}] = F(e^t)$.

Binomial moments via generating functions

The PGF for the binomial distribution is

$$F(t) = (pt+q)^n.$$

So

$$\begin{aligned}
\frac{d}{dt}F(t) &= np(pt+q)^{n-1}, \\
\frac{d^2}{dt^2}F(t) &= n(n-1)p^2(pt+q)^{n-2}, \\
&\cdots \\
\frac{d^k}{dt^k}F(t) &= n(n-1)(n-k+1)p^k(pt+q)^{n-k}.
\end{aligned}$$

Hence

$$E[X^{\underline{k}}] = n^{\underline{k}}p^k,$$

which is equivalent to

$$E\left[\binom{X}{k}\right] = \binom{n}{k}p^k.$$

Notes.

Lecture 3

Topics du jour: Binomial identities via lattice paths, discrete uniform distribution.

References: PB 3.5, GK 1.2.

Lattice paths

In an $m \times n$ lattice, there are $\binom{m+n}{n}$ direct paths between opposite corners.

By a direct path, we mean one that always goes across or up.

Proof: A direct path has $m+n$ edges, and we can choose the edges going across in n ways.

Connection to binomial identities

By cleverly thinking about these paths, we can prove various identities involving binomial coefficients.

Example 1: $\binom{n}{i} = \binom{n-1}{i-1} + \binom{n-1}{i}$

The last edge (which reaches B) can either go up or across.

Example 2: $\sum_{k=0}^{m} \binom{n+k}{k} = \binom{n+m+1}{m}$.

Consider an $n+1 \times m$ lattice, and group paths according to the last edge that goes up.

Example 3: $\sum_{i=0}^{r} \binom{r}{i}\binom{s}{n-i} = \binom{r+s}{n}$.

This is called *Vandermonde's convolution.*

Consider an $(r+s-n)\times n$ lattice. Draw the diagonal line $x+y=r$ in the lattice, and group paths according to where they cross the line. For $0 \le i \le r$, there will be a path through the lower left $i \times (r-i)$ box, followed by one through the upper right $(n-i)\times(s-n+i)$ box:

Discrete uniform distribution

Choose $X \in \{1,\ldots,n\}$ uniformly, so that $\Pr[X=i] = 1/n$ for $i = 1,\ldots,n$.

This gives the distribution of the time for successful search in a linear list of n keys, assuming that we are looking for a random key.

Mean and variance:

$$E[X] = \sum_{i=1}^{n} \frac{i}{n} = \frac{1}{n}\frac{n(n+1)}{2} = (n+1)/2.$$

$$E[X^2] = \sum_{i=1}^{n} \frac{i^2}{n} = \frac{1}{n}\frac{2n^3+3n^2+n}{6} = \frac{2n^2+3n+1}{6}.$$

$$\sigma^2[X] = E[X^2] - E[X]^2 = \frac{n^2-1}{12}.$$

Higher moments

Finding the k-th moment reduces to the calculation of $S_k(n) = \sum_{i=1}^{n} i^k$.

These sums can be handled by knowing that $S_k(n)$ is a polynomial in n, of degree $k+1$, and using small values of n to generate linear equations that can be solved for the coefficients.

Example. Above, we needed a formula for the sum of the first n squares. This works out best if we start the sum at 0, and use falling powers instead of regular powers. Let

$$S_2(n) = \sum_{i=0}^{n} i^2 = An^{\underline{3}} + Bn^{\underline{2}} + Cn^{\underline{1}} + D.$$

$S_2(0) = 0$, so $D = 0$.

$S_2(1) = 1 = C \cdot 1$, so $C = 1$.

$S_2(2) = 5 = B \cdot 2 + C \cdot 2 = 5$, so $B = 3/2$.

$S_2(3) = 14 = A \cdot 6 + B \cdot 6 + C \cdot 3$, so $A = 1/3$.

Expanding out the falling powers we get

$$S_2(n) = \frac{n^3}{3} + \frac{n^2}{2} + \frac{n}{6}.$$

How did we know to look for a polynomial? One way is to know the following result in the calculus of finite differences: the k-th difference of a sequence $\{u_i\}$ vanishes everywhere iff the sequence is given by a polynomial in i, of degree $k-1$.

The trick of using falling powers allows the coefficients to be found with $O(k^2)$ operations. If we used regular powers we would have a dense $k \times k$ system to solve, and standard methods for this (e.g. Gauss elimination) would take time $O(k^3)$.

You might wonder if there is a formula for the coefficients. This is a famous problem that was solved by Bernoulli.

Chained hashing

We will assume you are familiar with the method. For each item x, you choose a pseudorandom index i, and insert x onto the ith linked list. For details, see any book on data structures.

Let the number of lists be N, and assume we insert k keys into the table.

Our *model* will be that each key is inserted into a random (uniformly distributed) list. Note that this is really an assumption about the quality of the hash function.

What's the size of the j-th list?

For each j, the number of keys on the j-th list has the Binomial(k,$1/N$) distribution. Note that these are *not* independent.

So

$$E[\text{ list size }] = k/N,$$

and

$$\sigma^2[\text{ list size }] = k(1/N)(1-1/N) = k/N - k/N^2 \sim k/N$$

as $N \to \infty$.

We'll now study the expected time for successful search (the time to look up a random element x in the table).

The trick is to condition on the event

$$E_{ij} = \{x \text{ is on the } j\text{-th list, which has } i \text{ keys}\}.$$

Using results for the uniform distribution,

$$E\left[\#\text{ of probes}|E_{ij}\right] = \frac{i+1}{2}.$$

We also have

$$\Pr[E_{ij}] = \Pr[\ j\text{-th list has } i \text{ keys }] \times \Pr[\ x \text{ is on } j\text{-th list, given it has } i \text{ keys }].$$

$$= \binom{k}{i}\left(\frac{1}{N}\right)^i\left(1-\frac{1}{N}\right)^{k-i} \times \frac{i}{k}.$$

Combining the last two results, we have

$$\begin{aligned} E[\#\text{ of probes}] = & \sum_{\substack{1\le i\le k\\ 1\le j\le N}} E[\#\text{ of probes}|E_{ij}]\Pr[E_{ij}] \\ = & \sum_{\substack{1\le i\le k\\ 1\le j\le N}} \frac{i(i+1)}{2k}\binom{k}{i}\left(\frac{1}{N}\right)^i\left(1-\frac{1}{N}\right)^{k-i} \\ = & \frac{N}{2k}\sum_{1\le i\le k} i^2+i\binom{k}{i}\left(\frac{1}{N}\right)^i\left(1-\frac{1}{N}\right)^{k-i} \end{aligned}$$

So the expected number of probes is

$$\frac{N}{2k}E[X^2+X],$$

where X has the Binomial(k,$1/N$) distribution.

Using our formulas for the mean and standard deviation of the binomial distribution, we have

$$\begin{aligned} E[X^2+X] = & \ \sigma^2[X]+E[X]^2+E[X] \\ = & \ \frac{k}{N}(1-1/N)+k^2/N^2+k/N, \end{aligned}$$

so the expected number of probes is

$$\frac{N}{2k}\left(2k/N-k/N^2+k^2/N^2\right) = k/2N+(1-1/N).$$

Note that $k/2N$ is half the expected list size, so you can think of the second part of the formula as a correction to this crude guess.

Higher moments of the search time

Given any r, you can compute the r-th moment of the search time by the above technique, if you are sufficiently patient. (You will need to know binomial moments up to order $r+1$.)

Since we are not sufficiently patient, we will stop with $r=2$ and use this to get the variance.

We have

$$\begin{aligned} E[(\#\text{ of probes})^2] &= \sum_{i,j} E[\#\text{ of probes}|E_{ij}]\Pr[E_{ij}] \\ &= \sum_{i,j} \frac{2i^2+3i+1}{6}\binom{k}{i}\left(\frac{1}{N}\right)^i\left(1-\frac{1}{N}\right)^{k-i}\frac{i}{k} \\ &= \frac{N}{6k}E[2X^3+3X^2+X], \end{aligned}$$

and when $X \sim \text{Binomial}(n,p)$,

$$E[2X^3+3X^2+X] = 2n^3p^3 - 6n^2p^3 + 4np^3 + 9n^2p^2 - 9np^2 + 6np.$$

Plugging in $n=k$ and $p=1/N$, we find

$$E[(\#\text{ of probes})^2] = \frac{k^2}{3N^2} - \frac{k}{N^2} + \frac{2}{3N^2} + \frac{3k}{2N} - \frac{3}{2N} + 1.$$

So

$$\sigma^2[\#\text{ of probes}] = \frac{(k-1)(6N+k-5)}{12N^2}.$$

Naturally we have left out some steps. For problems like this (where you know what to do but there are many grundgy details) it makes sense to use a computer algebra system like Maple.

Comparison with crude model

After deriving a complicated result like this we should sit back and try to compare it with things we already know.

A naive way to think about chained hashing is that search takes place in a linked list of size equal to the expected value $\alpha = k/N$.

The crude model predicts

$$\text{mean} = \frac{\alpha+1}{2}, \qquad \text{variance} = \frac{\alpha^2-1}{12}.$$

(Already we have a problem when $\alpha \to 0$!)

For the exact model we have

$$\text{mean} = \frac{\alpha+1}{2} + \frac{1}{2} + O\left(\frac{1}{N}\right),$$

and you can think of the extra 1/2 as the result of *length-biased sampling*. (We are not picking something from a random list, we are picking from a list that we know has something on it!)

For fixed α and $N \to \infty$, the variance of the exact model is asymptotic to

$$\frac{\alpha^2 - 1}{12} + \frac{\alpha}{2} + \frac{1}{12},$$

and for small α the error represented by the last two terms can be significant. (Remember that for most hash tables, $\alpha < 1$.)

Notes

Moments of successful search time?

Lecture 4

Topics du jour: Occupancy

References: Feller 1 II.5, IV.2; David and Barton, *Combinatorial Chance,* p. 243 ff. The standard reference is V. Kolchin et al., *Random Allocations.*

The basic model

We drop k balls randomly and uniformly into N bins.

In general the placement will be irregular, and we want to know features of this placement such as:

How many bins are occupied (contain at least one ball)?

How many balls are in the bin with the largest (least) population?

This model has many applications, including:

Bucket sorting

Chained-bucket hashing

Birthday problems (when will some bin contain more than one ball?)

Statistical physics (where the model is called the Maxwell-Boltzmann distribution).

How many bins will be occupied?

We consider the number of empty bins, using the method of indicator variables.

Let $X_i = 1$ if bin i is empty, 0 if it is occupied.

The number of empty bins is

$$X = \sum_{i=1}^{N} X_i.$$

$E[X_i] = (1 - 1/N)^k$, so the expected number of empty bins is

$$E[X] = N(1 - 1/N)^k.$$

The expected number of occupied bins is

$$N - N(1 - 1/N)^k = N\left(1 - (1 - 1/N)^k\right).$$

A numerical approximation for $k = \alpha N$:

$$(1 - 1/N)^k = (1 - 1/N)^{\alpha N} \sim e^{-\alpha}$$

So the mean fraction of occupied bins is asymptotic to

$$1 - e^{-\alpha}$$

when α is fixed and $N \to \infty$.

Higher moments can be computed but we need another trick.

As an application for this problem, consider chained hashing where the chains are stored in external memory, on pages. Suppose the pages are large enough that the chance of overflowing a page can be neglected. The number of pages used is the number of occupied bins.

Another application: size of "working sets" in memory management.

Vandermonde's theorem:

Let $n \geq 0$ be an integer. Then $(x+y)^{\underline{n}} = \sum_i \binom{n}{i} x^{\underline{i}} y^{\underline{n-i}}$.

This is like the binomial theorem, but it uses falling powers rather than regular powers.

Can be proved by induction on n.

The cases $n = 0, 1$ can be verified directly, using the laws $x^{\underline{0}} = 1$ and $x^{\underline{1}} = x^1 = x$.

To go from n to $n+1$, compute

$$\begin{aligned}
(x+y)^{\underline{n+1}} &= (x+y)^{\underline{n}}(x+y-n) \\
&= \sum_i \binom{n}{i} x^{\underline{i}} y^{\underline{n-i}} ((x-i)+y-(n-i)) \\
&= \sum_i \binom{n}{i} x^{\underline{i+1}} y^{\underline{n-i}} + \sum_i \binom{n}{i} x^{\underline{i}} y^{\underline{n-i+1}} \\
&= \sum_i \binom{n}{i-1} x^{\underline{i}} y^{\underline{n-i+1}} + \sum_i \binom{n}{i} x^{\underline{i}} y^{\underline{n+1-i}} \\
&= \sum_i \binom{n+1}{i} x^{\underline{i}} y^{\underline{n+1-i}}.
\end{aligned}$$

There is also a combinatorial proof. Fill an urn with x red balls and y white balls. The left side is the number of ways to choose a sequence of n balls. The right side corresponds to an experiment in which we first decide the color for each position in the sequence, and then fill the red (resp. white) positions one by one.

Multinomial version:

$$(x_1 + \ldots + x_n)^{\underline{m}} = \sum_\alpha \binom{m}{\alpha} x_1^{\underline{\alpha_1}} \ldots x_n^{\underline{\alpha_n}}.$$

Here, as in the multinomial theorem,

$$\alpha = (\alpha_1, \ldots, \alpha_n)$$

is a so-called *multi-index*, and we sum over the multi-indices with components between 0 and n, which sum to m.

This can be proved by induction on n (I think). The combinatorial proof extends to this case as well. Left as exercises for you.

Computing moments of indicator sums

This idea is due to M. Fréchet.

Let $E_1, \ldots, E_n$ be events (need not be independent).

Let X_i be an variable indicating whether E_i happens, i.e.,

$$X_i = \begin{cases} 1, & \text{if } E_i \text{ occurs;} \\ 0, & \text{otherwise.} \end{cases}$$

and put $X = \sum_i X_i$.

By Vandermonde's theorem

$$X^{\underline{m}} = (X_1 + \ldots + X_n)^{\underline{m}} = \sum_{\alpha} \binom{m}{\alpha} X_1^{\underline{\alpha_1}} \ldots X_n^{\underline{\alpha_n}}$$

We have $X_i \in \{0, 1\}$, so the terms in which $\alpha_i \geq 2$ will vanish. So we only have to sum over multi-indices in which the components are 0 or 1. Then only exponents of 0 or 1 remain, and we can also replace the falling powers by ordinary powers. Doing this, we get

$$X^{\underline{m}} = \sum_{\substack{0 \leq \alpha_i \leq 1 \\ \Sigma \alpha_i = m}} m! X_1^{\alpha_1} \ldots X_n^{\alpha_n} = m! \sum_{i_1 < i_2 < \cdots < i_m} X_{i_1} X_{i_2} \cdots X_{i_m}.$$

Taking expected values, we find

$$E\left[X^{\underline{m}}\right] = m! \sum_{i_1 < i_2 < \cdots < i_m} \Pr[E_{i_1} \cap E_{i_2} \cap \cdots \cap E_{i_m}].$$

This is most useful when the events are symmetric, in the sense that the probabilities in question are all the same.

Moments of the occupancy distribution

We use Fréchet's trick, with E_i denoting the event that the i-th bin is empty. Then X is the number of empty bins.

The chance that any given m bins are all empty is

$$(1 - m/N)^k .$$

There are $\binom{N}{m}$ ways to choose these m bins, so

$$E\left[X^{\underline{m}}\right] = m!\binom{N}{m}(1 - m/N)^k = N^{\underline{m}}\left(1 - \frac{m}{N}\right)^k .$$

We get the variance of X from the formula for $m = 2$.

$$\begin{aligned}\sigma^2[X] = &\ E[X(X-1)] + E[X] - E[X]^2 \\ = &\ N(N-1)(1-2/N)^k + N(1-1/N)^k - N^2(1-1/N)^{2k}\end{aligned}$$

As with the mean, we can obtain an approximation when $\alpha = k/N$ is fixed and $N \to \infty$. First, rewrite the variance as

$$N^2\left[(1-2/N)^k - (1-1/N)^{2k}\right] - N(1-2/N)^k + N(1-1/N)^k.$$

The last two terms are easy to handle, but the first requires some care. We have, by applying the mean value theorem to $f(z) = (1 - z/N)^k$,

$$\begin{aligned}&N^2\left[(1-2/N)^k - (1-1/N)^{2k}\right] \\ =&N^2\left[(1-2/N)^k - (1-(2-1/N)/N)^k\right] \\ =&-k\left(1-2/N+O(N^{-2})\right)^{k-1} \qquad \text{[by mean value thm]} \\ \sim&-\alpha N e^{-2\alpha} \qquad \text{[using the series for } \log(1-z)\text{]}.\end{aligned}$$

Bottom line: when the load factor $\alpha = k/N$ is constant, the number of occupied bins has

$$\text{mean } \sim N(1-e^{-\alpha}), \qquad \text{variance } \sim N(e^{-\alpha} - e^{-2\alpha} - \alpha e^{-2\alpha}).$$

The mean is $\Theta(N)$, but the standard deviation is only $O(\sqrt{N})$.

It can be shown that for this regime (α constant), the limiting distribution is normal. (See David and Barton.)

Notes

This should be hooked up to the Poisson process model (random number of balls) in some fashion.

The "sampling without replacement" model is interesting too. The expectation was found by S.P Yao, CACM v. 20, no. 4, 1977, pp. 260-261.

Lecture 5

Topics du jour: Maximum occupancy (longest chain length), geometric distribution (repeating random hashing).

References: For occupancy: K. Mehlhorn, *Data Structures and Algorithms*, v. 1. G. Gonnet, J. ACM, 1981. For hashing: PB, Feller v. 1.

Review of occupancy model

N bins into which we throw k balls.

Balls are indistinguishable, so we can only ask about the number of balls in each bin.

Last lecture: studied the distribution of the number of occupied bins.

Maximum occupancy

One of the bins will have more (or at least no fewer) balls than any other. How many balls are in this "most crowded" bin?

Corresponds to the following pessimistic question about chained hashing: After the keys are put into the table, how long do we think the longest search will take?

Note that this is not a question about the absolute worst case, but about the worst behavior a user might reasonably expect to encounter. We will answer this by proving a *large deviation* bound, which says that (in a certain sense) the chance of a slow search is very small.

Large deviation bounds for maximum occupancy in hashing

Number of balls in a bin is binomially distributed

More precisely, Binomial(k,$1/N$).

When $\alpha = k/N$ is fixed, a standard approximation to the binomial distribution is given by the *Poisson* distribution, whose probabilities are

$$p_i = \frac{e^{-\alpha}\alpha^i}{i!}, \qquad i = 0, 1, 2, \ldots$$

For our purposes, a crude form of this approximation is good enough.

A bound on the binomial probability

$$\binom{k}{i} N^{-i}(1 - 1/N)^{k-i} \leq \binom{k}{i} N^{-i} \leq \frac{\alpha^i}{i!} \leq \frac{1}{i!}$$

Bounds on the chance of a large chain

The probability that there is a chain of length $\geq m$ is at most

$$N\Pr[\text{ a given chain is } \geq m\ ] \leq N \sum_{i \geq m} \frac{\alpha^i}{i!}.$$

Choose m_0 to be the smallest integer satisfying $m_0!/\alpha^{m_0} \geq N$. Using Stirling's formula it is possible to show that

$$m_0 \sim \frac{\log N}{\log\log N}$$

(Proof: Since α is fixed, we have $\log N \sim \log m_0! \sim m_0 \log m_0$. Taking logs again, we find $\log m_0 \sim \log\log N$, so $m_0 \sim (\log N)/(\log m_0) \sim (\log N)/(\log\log N)$.)

Now let m_1 be the least integer satisfying

$$m_1 \geq m_0, \qquad \frac{\alpha}{m+1} \leq \frac{1}{2}.$$

Since the second requirement knocks out only a finite number of candidates, we also have

$$m_1 \sim \frac{\log N}{\log\log N}$$

By our crude Poisson estimate, the chance there is any chain of length $\geq m_1 + t$ is at most

$$\begin{aligned} N \sum_{i \geq m_0 + t} \frac{\alpha^i}{i!} &\leq N\left[\frac{\alpha^{m_1+t}}{(m_1+t)!} + \frac{\alpha^{m_1+t+1}}{(m_1+t+1)!} + \cdots\right] \\ &\leq N\left[\frac{\alpha^t}{(m_1+1)^t} + \frac{\alpha^{t+1}}{(m_1+1)^{t+1}} + \cdots\right] \\ &\leq N\left[\frac{1}{2^t} + \frac{1}{2^{t+1}} + \cdots\right] \\ &= \frac{1}{2^{t-1}}. \end{aligned}$$

This shows that once we reach the critical value $m_1 \sim (\log N)/(\log\log N)$, the probability of a larger chain goes down exponentially.

Expected maximum occupancy

From the tail bound we proved it follows that

$$E[\text{ max chain length }] \leq m_1 + O(1) \sim \frac{\log N}{\log\log N}.$$

G. Gonnet (J. ACM, v. 28, 1981) gives the following formula: for $\alpha = k/N$ fixed and $N \to \infty$,

$$E[\text{ maximum chain size }] = \Gamma^{-1}(N)\left(1 + \frac{\log \alpha}{\log \Gamma^{-1}(N)} + O((\log \Gamma^{-1}(N))^{-2})\right).$$

Here $\Gamma(x)$ is defined by

$$\Gamma(x) = \int_0^\infty t^{x-1} e^{-t} dt.$$

This is a continuous extension of the factorial function, i.e. if x is a non-negative integer, we have

$$\Gamma(x+1) = x!.$$

As before, we can use Stirling's formula to prove $\Gamma^{-1}(N) \sim \frac{\log N}{\log \log N}$.

It is interesting that the load factor α only affects the second order term.

In-place hashing.

The table is an array indexed by $1, \ldots, N$. Each key k determines a (pseudorandom) *probe sequence*

$$x_1, x_2, \ldots$$

of addresses. To find a key, you follow its probe sequence until you find they key or you have looked at all places that it could be. Insertion works the same way except that you are looking for an empty slot into which to put the key.

This generated a huge amount of research in the early days of computer science. Less important now because memory is not so tight.

Exact analysis of explicit algorithms (e.g. linear probing) is hard.

We will use two simplified models: a) the probe sequence is random and b) the probe sequence is a permutation of $\{1, \ldots, N\}$. These correspond to sampling an urn with and without replacement. The justification is that probe sequences are designed to act like random address sequences.

Geometric distribution

Let $X_1, X_2, \ldots$ be i.i.d. Bernoulli trials, with success probability p (and failure probability $q = 1 - p$).

If T is the first index t with X_t a success, then we say that T has a *geometric* distribution.

We have

$$\Pr[T = t] = q^{t-1} p.$$

Warning: some authors count the number of trials before the first success, rather than the number of trials needed for a success. Our definition is better for analyzing algorithms.

Moments:

The expected value is easy to compute. We must do one trial, and after that there is a chance q that the process will repeat, behaving in an identical way. So

$$E[T] = 1 + qE[T],$$

which can be solved to get

$$E[T] = \frac{1}{1-q} = \frac{1}{p}.$$

Higher moments can be obtained by summing series. It is convenient to use falling powers. For example,

$$\begin{aligned} E[T^{\underline{2}}] &= \sum_{k\geq 0} k(k-1)q^{k-1}p \\ &= pq\sum_{k\geq 0} k(k-1)q^{k-2} \\ &= pq\frac{d^2}{dq^2}(1-q)^{-1} \\ &= \frac{2q}{p^2}. \end{aligned}$$

Similarly,

$$E[T^{\underline{k}}] = \frac{k!q^{k-1}}{p^k}.$$

We can obtain the variance as

$$E[T(T-1)] + E[T] - E[T]^2 = \frac{q}{p^2}.$$

Bottom line:

$$E[T] = \frac{1}{p} \qquad \text{exactly}$$

and

$$\sigma^2[T] \leq \frac{1}{p^2}.$$

When p is small, the mean and standard deviation are approximately equal. This means that we can expect to see rather large fluctuations in practice. (Keep this in mind the next time you are waiting for something to happen!)

Application to in-place hashing.

Suppose k out of N table entries are occupied. If we model probes as i.i.d. uniform addresses, the number of probes in an insertion is geometric with

$$p = \frac{N-k}{N}, \qquad q = \frac{k}{N}.$$

For hashing it is traditional to use the load factor $\alpha = k/N$ as the parameter.

Expected number of probes for insertion is

$$\frac{N}{N-k} = \frac{1}{1-\alpha}.$$

Variance in the number of probes is

$$\frac{q}{p^2} = \frac{\alpha}{(1-\alpha)^2}.$$

The i.i.d. uniform model is a bit unrealistic, and we should think about ways it can be improved. Here is one idea. It does not make sense to probe any cell that we have already seen, and most pseudo-random hashing schemes prevent this. Let us model this by assuming that the next probe is a uniform random draw from the cells we have not yet seen. Call the corresponding waiting time U. Our intuition suggests that U should be "smaller" than T.

$U \le N$, whereas T is unbounded (although it is finite with probability 1).

We can find a common sample space on which $U \le T$. Consider the space of i.i.d. random samples

$$X_1, X_2, X_3, \ldots,$$

from $\{1, \ldots, N\}$. We have

$$T = \text{ least } i \text{ with } X_i \le Np$$

whereas

$$U = T - \text{ \# of repeats in } X_1, \ldots, X_{T-1} \ .$$

Therefore, $E[T] \le E[U]$ and in general $E[T^m] \le E[U^m]$ for every $m \ge 0$.

Notes

Good exercise: look at what happens when the load factor grows with n, say $\alpha = \log n$. Maximum should be $O(\log n)$ (Nishimura). Kolchin et al., Random Allocations, gives a limiting distribution for the maximum in this case.

More results on the maximum chain size can be found in Kolchin.

David and Barton (p. 279) have the following formula. For r balls into m boxes, the probability that no box has b or more balls is

$$\frac{1}{m^r} \times \text{ coeff of } x^r/r! \text{ in the expansion of } \left(\sum_{i=0}^{b-1} x^i/i!\right)^m \ .$$

This is of course the probability that the longest chain is $< b$.

Lecture 6

Topics du jour: Hypergeometric waiting time (model for nonrepeating random hashing), introduction to hypergeometric functions.

Waiting time distributions

We will describe these with an urn model. Suppose we have N balls total, with Np red and Nq white (so that $p + q = 1$).

Geometric waiting time: Number of samples *with replacement* needed to obtain a red ball. Call this random variable T.

Hypergeometric waiting time: Number of samples *without replacement* needed to obtain a red ball. Call this random variable U.

We observed last time that there is a natural way to define U on T's sample space, and there $U \leq T$.

Expected value of hypergeometric waiting time

The following clever argument comes from F. Mosteller, *Fifty Challenging Problems in Probability*, p. 41.

Let there be r red balls, w white ones. Arrange all the balls into a random order:

$$R\ W\ R\ W\ W$$

Now add one more red ball at the end, and wrap them around into a circle:

$$\begin{array}{ccc} & R & \\ W & & R \\ W & & W \\ & R & \end{array}$$

Going clockwise from the top, we have $r + 1$ successive instances of W^*R. (Computer science jargon for "zero or more W, followed by one R.") Call their lengths $U_1, U_2, \ldots, U_{r+1}$. Evidently the U_i have the same distribution. So

$$E[U] = E[U_1] = \frac{1}{r+1} \sum_{i=1}^{r+1} E[U_i] = \frac{1}{r+1} E[\sum_{i=1}^{r+1} U_i] = \frac{N+1}{r+1}.$$

You can check that $E[U] \leq N/r$.

Note that the expected waiting time depends both on the fraction of red balls and on the total number of balls in the urn. As $N \to \infty$ (with α fixed) the second effect dies out and the model acts more and more like like sampling with replacement.

Application to hashing

Assume the probe sequence is a random permutation of $1, \ldots, N$. We analyze the number of probes needed to find an empty slot (this corresponds to inserting a new element into our table).

Let there be k slots full, $r = N - k$ free. The expected number of probes is

$$\frac{N+1}{N-k+1} = \frac{1+1/N}{(1-\alpha)+1/N}.$$

(Here $\alpha = k/N$ as usual.)

We have not yet derived an exact formula for the variance, but we can easily estimate it as follows:

$$\sigma^2(U) = E[U^2] - E[U]^2 \le E[T^2] - E[U]^2 = \frac{N(N+k)}{(N-k)^2} - \frac{(N+1)^2}{(N-k+1)^2}.$$

For fixed α and $N \to \infty$ this estimate has the limit

$$\frac{1+\alpha}{(1-\alpha)^2} - \frac{1}{(1-\alpha)^2}\frac{\alpha}{(1-\alpha)^2},$$

which is the variance for repeating random hashing.

Another approach to the mean waiting time

Since this is a "methods" course, we will compute the expected value of U in another way. Besides giving you another technique for your toolbox, this way is more powerful and will ultimately lead to a way to find any moment of U.

The probability that t samples from the urn fail to find a red ball is the number of bad sample sets divided by the total number of sample sets, that is

$$\frac{\binom{N-r}{t}}{\binom{N}{t}} = \frac{(N-r)^{\underline{t}}}{N^{\underline{t}}}.$$

So the expected value of U is

$$E[U] = \sum_{t \ge 0} \Pr[U > t] = \sum_{t \ge 0} \frac{(N-r)^{\underline{t}}}{N^{\underline{t}}}.$$

This is a more complicated sum than we have yet seen, but it can be evaluated by the method of hypergeometric functions. Recall that

$$\sum_i \binom{n}{i} = 2^n,$$

because the function $(1+x)^n$ evaluates to 2^n at $x=1$. We want something that can play the role of $(1+x)^n$ for more complicated combinatorial sums. This "something" will turn out to be a hypergeometric function, to whose properties we now turn.

What are hypergeometric functions?

Defn. A *hypergeometric series* is something of the form

$$F(z) = \sum_{k\geq 0} c_k z^k$$

in which the coefficient ratio c_{k+1}/c_k is a rational function of k.

Example: The exponential function has the series

$$\exp(z) = 1 + z + z^2/2 + \ldots + z^k/k! + \ldots,$$

in which $A_{k+1}/A_k = 1/(k+1)$.

We will be interested in

$${}_mF_n = \left(\begin{matrix} a_1, a_2, \ldots, a_m \\ b_1, b_2, \ldots, b_n \end{matrix}; z\right) = \sum_{k\geq 0} \frac{a_1^{\overline{k}} \cdots a_m^{\overline{k}}}{b_1^{\overline{k}} \cdots b_n^{\overline{k}}} \frac{z^k}{k!},$$

in which

$$a^{\underline{k}} = a(a+1)...(a+k-1).$$

denotes the *k-th rising power of a*.

Note that $k! = 1^{\overline{k}}$.

Also $a^{\underline{k}} = (-1)^k(-a)^{\overline{k}}$.

Many of the common functions of analysis are expressible by hypergeometic series.

$${}_0F_0(z) = \exp(z)$$

$${}_1F_0\left(\begin{matrix} a \\ \end{matrix}; z\right) = (1-z)^{-a}$$

${}_1F_1$ is usually called the *confluent hypergeometric function*. For example,

$${}_1F_1\left(\begin{matrix} 1 \\ 2 \end{matrix}; z\right) = \frac{e^z - 1}{z}.$$

${}_2F_1$ is the *Gaussian hypergeometric function* (some people call this "the" hypergeometric function). For example,

$${}_2F_1\left(\begin{matrix} 1, 1 \\ 2 \end{matrix}; z\right) = -\frac{\log(1-z)}{z}.$$

Analytic properties:

When $m \le n$, ${}_mF_n$ converges everywhere (bound by $\exp(z)$).

When $m = n+1$, ${}_mF_n$ converges whenever $|z| < 1$ (bound by a geometric series).

If one of the $a_i = -d$ (negative integer), ${}_mF_n$ is a polynomial of degree d.

For fixed z, ${}_mF_n$ is a continuous function of its parameters.

A standard reference on these functions is L. J. Slater, *Generalized Hypergeometric Functions,* Cambridge U. Press 1966.

Special values of hypergeometric series.

Chu-Vandermode:

$$ {}_2F_1\left(\begin{matrix}-a,-n\\ c\end{matrix};1\right) = \frac{(a+c)^{\overline{n}}}{(c)^{\overline{n}}}. $$

Gauss:

$$ {}_2F_1\left(\begin{matrix}-a,-b\\ c\end{matrix};1\right) = \frac{\Gamma(a+b+c)\Gamma(c)}{\Gamma(a+c)\Gamma(b+c)}. $$

Here c is not 0 or a negative integer, and $a+b+c > 0$.

The hypergeometric method for combinatorial sums

The basic idea is easy to state. The hypergeometic series gives a standard form for a large class of sums involving binomial coefficients, factorials, etc. We reduce the sum to this standard form and see if it is a known value.

Example: expected hypergeometric waiting time.

We have

$$ E[U] = \sum_{t\ge 0} \frac{(N-r)^{\underline{t}}}{N^{\underline{t}}} = \sum_{t\ge 0} \frac{(r-N)^{\overline{t}}\ 1^{\overline{k}}}{(-N)^{\overline{t}}\ t!} 1^t = {}_2F_1\left(\begin{matrix}1, r-N\\ -N\end{matrix};1\right) $$

Taking $a = -1$, $n = N - r$, and $c = -N$ in the Chu-Vandermonde identity, we find

$$ E[U] = \frac{(-1-N)^{\overline{N-r}}}{(-N)^{\overline{N-r}}} = \frac{(N+1)^{\underline{N-r}}}{(N)^{\underline{N-r}}} = \frac{N+1}{r+1}. $$

Notes.

For more on hypergeometric waiting times, see

S.S. Wilks, Mathematical Statistics, pp. 141-143.

Kemp and Kemp, J. Royal Statist. Soc. B, v. 18, pp. 202-211.

For lists of functions that can be expressed "hypergeometrically," see

Lebedev, pp. 271–274 (various ${}_1F_1$).

Abramowitz and Stegun, pp. 556, 561 (various ${}_2F_1$).

Lebedev, pp. 258–260 (various ${}_2F_1$).

Askey, Orthogonal Polynomials and Special Functions [SIAM 1975] (various orthogonal polynomials, plus Bessel functions).

Andrews, Askey, and Roy, Special Functions, pp. 102-107 (dilogarithms),

Abramowitz and Stegun, pp. 945 ff. (various probability functions).

Whittaker and Watson p. 499 (elliptic function periods).

Lecture 7

Topics du jour: Combinatorial sums via hypergeometric functions.

References: PB 3.8, R. Roy, AMM 1987. For general facts about hypergeometric functions see Andrews, Askey, and Roy, *Special Functions*.

What's this about?

In analyzing algorithms we are often confronted with sums involving binomial coefficients. We will present a general method for attacking these, the philosophy of which can be summed up as

$$\text{binomial identities} \equiv \text{special values of hypergeometric functions}$$

Notation:

k-th rising power of a:

$$a^{\overline{k}} = a(a+1)...(a+k-1).$$

We can convert from rising to falling powers using

$$a^{\underline{k}} = (-)^k(-a)^{\overline{k}}$$

Generalized hypergeometric function

$${}_mF_n = \left(\begin{matrix} a_1, a_2, \ldots, a_m \\ b_1, b_2, \ldots, b_n \end{matrix}; z\right) = \sum_{k\geq 0} \frac{a_1^{\overline{k}} \cdots a_m^{\overline{k}}}{b_1^{\overline{k}} \cdots b_n^{\overline{k}}} \frac{z^k}{k!} = 1 + \frac{a_1 \cdots a_m}{b_1 \cdots b_n} z + \cdots$$

Gamma function:

$$\Gamma(s) = \int_0^\infty x^{s-1} e^{-x} dx, \qquad s > 0.$$

For integer $n \geq 1$, $\Gamma(n) = (n-1)!$.

Extend to non-integer values and 0 via $\Gamma(s) = s^{-1}\Gamma(s+1)$.

Special values of hypergeometric functions

For this lecture we will need the Chu-Vandermonde identity:

$${}_2F_1\left(\begin{matrix} -n, b \\ c \end{matrix}; 1\right) = \frac{(c-b)^{\underline{n}}}{c^{\underline{n}}}$$

Several more results of this type are listed in an appendix.

Examples of hypergeometric summation.

1) What is

$$\sum_{k=0}^{n} \binom{n}{k}^2 ?$$

We convert the sum to rising powers:

$$\sum_{k=0}^{n} \binom{n}{k}^2 = \sum_{k=0}^{n} \frac{(n^{\underline{k}})^2}{(k!)^2} = \sum_{k=0}^{n} \frac{(-n)^{\overline{k}}(-n)^{\overline{k}}}{1^{\overline{k}}} \frac{1^k}{k!} = {}_2F_1\left(\begin{matrix}-n,-n\\1\end{matrix};1\right).$$

Taking $b = -n$ and $c - 1$ in Chu-Vandermonde, this is

$$\frac{(n+1)^{\overline{n}}}{1^{\overline{n}}} = \frac{(n+1)(n+2)\cdots 2n}{n!} = \binom{2n}{n}.$$

A probabilistic interpretation: Consider an urn with n red balls and n white balls. We draw (without replacement) n random balls. The chance that exactly k of them are red is

$$\frac{\binom{n}{k}\binom{n}{n-k}}{\binom{2n}{n}} = \frac{\binom{n}{k}^2}{\binom{2n}{n}}.$$

The identity we just proved states that these probabilities sum to 1.

A lattice path interpretation: Consider corner-to-corner paths in an $n \times n$ lattice, and group them according to where they cross the line $x + y = n$.

2) [Knuth I p. 59] What is

$$\sum_{k\geq 0} \binom{n+k}{2k}\binom{2k}{k}\frac{(-1)^k}{k+1}?$$

Replace the binomials by factorials, and observe that $(2k)!$ cancels. This gives

$$\sum_{k\geq 0} \frac{(n+k)!(-1)^k}{(n-k)!k!(k+1)!}.$$

Since

$$\frac{a^{\overline{k+1}}}{a^{\overline{k}}} = k + a,$$

we can read the parameters off the term ratio. Letting A_k be a term of the sum, we have

$$\frac{A_{k+1}}{A_k} = (-1) \times \frac{(n-k)!k!(k+1)!(n+k+1)!}{(n+k)!(n-k-1)!(k+1)!(k+2)!} = \frac{(k-n)(k+n+1)}{(k+2)(k+1)}$$

So the sum is

$${}_2F_1\left(\begin{matrix}-n, n+1\\ 2\end{matrix}; 1\right) = \frac{(1-n)^{\overline{n}}}{2^{\overline{n}}} = \begin{cases} 0, & \text{if } n > 0; \\ 1, & \text{if } n = 0. \end{cases}$$

Variance for nonrepeating random hashing

Recall the following hashing model: we have N cells, some of which are free. To insert a new record, sample cells without repetition until we find one that is free.

This is equivalent to the following urn problem. Start with N balls, of which r are red and w white. Let U denote the number of draws needed to obtain a red ball. We showed previously that

$$E[U] = \frac{N+1}{r+1}.$$

Our goal is now to get the variance of U.

For $k \geq 1$ we have

$$\begin{aligned} p_k := \Pr[U = k] = \Pr[\text{ white}^{k-1}\text{red }] &= \frac{w}{N}\frac{w-1}{N-1}\cdots\frac{w-k+2}{N-k+2}\frac{r}{N-k+1} \\ &= \frac{r}{w+1}\frac{(w+1)^{\underline{k}}}{N^{\underline{k}}} \\ &= \frac{r}{w+1}\frac{(-1-w)^{\overline{k}}1^{\overline{k}}}{(-N)^{\overline{k}}\ k!}. \end{aligned}$$

Let

$$G(z) = \sum_{k\geq 0} p_k z^k.$$

Note that (and this has nothing to do with the form of G, it is valid in general)

$$G'(z) = \sum_{k\geq 0} k p_k z^k \qquad \Rightarrow \qquad E[U] = G'(1),$$

and

$$G''(z) = \sum_{k\geq 0} k(k-1) p_k z^k \qquad \Rightarrow \qquad E[U(U-1)] = G''(1).$$

Our $G(z)$ is essentially hypergeometric:

$$G(z) = \frac{r}{w+1}\ {}_2F_1\left(\begin{matrix}-1-w, 1\\ -N\end{matrix}; 1\right) + \text{const.}$$

(The exact constant is irrelevant since we will soon differentiate.)

From the definition we have

$$\frac{d}{dz}\ {}_2F_1\left(\begin{matrix}a, b\\ c\end{matrix}; z\right) = \frac{ab}{c}\ {}_2F_1\left(\begin{matrix}a+1, b+1\\ c+1\end{matrix}; z\right).$$

You can see that with sufficient patience we can express any desired moment of U in terms of hypergeometric function values. Since we are not sufficiently patient, we will just do the second moment. This requires

$$\frac{d^2}{dz^2}\ {}_2F_1\left(\begin{matrix}a, b\\ c\end{matrix}; z\right) = \frac{a(a+1)b(b+1)}{c(c+1)}\ {}_2F_1\left(\begin{matrix}a+2, b+2\\ c+2\end{matrix}; z\right).$$

Applying this formula we have

$$\begin{aligned} G''(z) &= \frac{r}{w+1}\frac{(-1-w)(-w)\cdot 1\cdot 2}{(-N)(-N+1)}\ {}_2F_1\left(\begin{matrix}1-w, 3\\ -N+2\end{matrix}; z\right) \\ &= \frac{2rw}{N(N-1)}\ {}_2F_1\left(\begin{matrix}1-w, 3\\ -N+2\end{matrix}; z\right). \end{aligned}$$

From Chu-Vandermonde with $n = w-1$, $b = 3$, $c = -N+2$, we get

$$G''(1) = \frac{2rw}{N(N-1)}\frac{(-N-1)^{\overline{w-1}}}{(-N+2)^{\overline{w-1}}} = \frac{2rw}{N(N-1)}\frac{(N+1)^{\underline{w-1}}}{(N-2)^{\underline{w-1}}}.$$

Expanding the falling powers, many factors cancel and we get (since $r+w=N$)

$$E[U(U-1)] = G''(1) = \frac{2(N-r)(N+1)}{(r+2)(r+1)}.$$

Finally, the variance:

$$\begin{aligned} \sigma^2(U) &= E[U(U-1)] + E[U] - E[U]^2 \\ &= \frac{2(N-r)(N+1)}{(r+2)(r+1)} + \frac{N+1}{r+1} + \frac{(N+1)^2}{(r+1)^2} \\ &= \frac{(N+1)}{(r+1)^2(r+2)}\left[2(N-r)(r+1) + (r+1)(r+2) - (N+1)(r+2)\right]. \end{aligned}$$

The expression in brackets is a quadratic polynomial in r with lead coefficient -1, which vanishes when $r=0$ (plug in) and $r=N$ (no white balls). So it is $r(N-r)$.

$$\sigma^2(U) = \frac{r(N-r)(N+1)}{(r+1)^2(r+2)}.$$

This should be compared with repeating hashing.

APPENDIX

I learned the procedure below from Dick Askey. The following four steps will determine if a sum of the form

$$\sum_{k\geq 0} A_k$$

is hypergeometric, and if so, determine the parameters and argument.

1) By removing common factors if necessary, normalize the sum so that $A_0 = 1$.
2) Compute the term ratio A_{k+1}/A_k. This should be a rational function of k.
3) Factor the term ratio as

$$\frac{A_{k+1}}{A_k} = \frac{\prod_{i=1}^{m}(k+a_i)}{\prod_{i=1}^{n}(k+b_i)(k+1)}z.$$

(By adding an extra factor upstairs if necessary, we can always make the last lower factor be $k+1$.)

4) Then

$$\sum_{k\geq 0} A_k = {}_mF_n\left(\begin{matrix}a_1,\ldots,a_m\\ b_1,\ldots,b_n\end{matrix};z\right).$$

if you are lucky it is a known value.

Recognizing Summable Hypergeometric Values

Many special values of hypergeometric series are known. Basically, any ${}_2F_1(1)$ can be summed, but for more complicated series the parameters must satisfy certain conditions. The easiest conditions to understand says that the row sums are nearly equal:

${}_mF_n$ is *balanced* if $1 + \sum a_i = \sum b_i$.

${}_mF_n$ is *k-balanced* if $k + \sum a_i = \sum b_i$.

We can place similar conditions on column sums. These make more sense if you put the parameters in an array, with an extra 1 (coming from $k!$) below:

$$\begin{pmatrix} a_1 & a_2 & \cdots & a_m \\ 1 & b_1 & \cdots & a_n \end{pmatrix}$$

Then

${}_mF_n$ is *well-poised* if the column sums are constant.

A well-poised series is called *very well-poised* if the second column has difference 1. This implies that the array must look like this:

$$\begin{pmatrix} a_1 & a_2+1 & \cdots \\ 1 & a_21 & \cdots \end{pmatrix}$$

From the first column sum we get $a_1 + 1 = 2a_2 + 1$, so $a_1 = 2a_2$.

Here are some useful identities. Parameters that can be filled in from the conditions are indicated with a star.

1) ${}_2F_1(1)$, any parameters [Gauss]:

$$ {}_2F_1\left(\begin{matrix}a,b\\c\end{matrix};1\right) = \frac{\Gamma(c)\Gamma(c-a-b)}{\Gamma(c-a)\Gamma(c-b)} $$

2) ${}_2F_1(1)$, with $-n$ on top: [Chu-Vandermonde]:

$$ {}_2F_1\left(\begin{matrix}-n,b\\c\end{matrix};1\right) = \frac{(c-b)^{\overline{n}}}{c^{\overline{n}}} $$

3) ${}_3F_2(1)$, $-n$ on top, balanced ($\sum \text{top} + 1 = \sum \text{bottom}$) [Saalschütz]:

$$ {}_3F_2\left(\begin{matrix}-n,a,b\\c,*\end{matrix};1\right) = \frac{(c-a)^{\overline{n}}(c-b)^{\overline{n}}}{c^{\overline{n}}(c-a-b)^{\overline{n}}} $$

4) ${}_3F_2(1)$, well-poised ($a_1 + 1 = a_2 + b_1 = ... = b_{m-1} + a_n$) [Dixon]:

$$ {}_3F_2\left(\begin{matrix}a & b & c\\ & * & *\end{matrix};1\right) $$

$$ = \frac{\Gamma(a+1-b)\Gamma(a+1-c)\Gamma(a/2+1)\Gamma(a/2+1-b-c)}{\Gamma(a/2+1-b)\Gamma(a/2+1-c)\Gamma(a+1)\Gamma(a+1-b-c)} $$

5) ${}_5F_4(1)$, well-poised with one column difference of 1 ("very well poised") [Dougall]:

$$ {}_5F_4\left(\begin{matrix}a & a/2+1 & b & c & d\\ & a/2 & * & * & *\end{matrix};1\right) $$

$$ = \frac{\Gamma(a+1-b)\Gamma(a+1-c)\Gamma(a+1-d)\Gamma(a+1-b-c-d)}{\Gamma(a+1)\Gamma(a+1-b-c)\Gamma(a+1-b-d)\Gamma(a+1-c-d)} $$

6) ${}_7F_6(1)$, very well poised and 2-balanced ($\sum \text{top} + 2 = \sum \text{bottom}$) [Dougall]:

$$ {}_7F_6\left(\begin{matrix}a & a/2+1 & b & c & d & * & -n\\ & a/2 & * & * & * & * & *\end{matrix};1\right) $$

$$ = \frac{(a+1)^{\overline{n}}(a+1-b-c)^{\overline{n}}(a+1-b-d)^{\overline{n}}(a+1-c-d)^{\overline{n}}}{(a+1-b)^{\overline{n}}(a+1-c)^{\overline{n}}(a+1-d)^{\overline{n}}(a+1-b-c-d)^{\overline{n}}} $$

Notes

Certain ${}_3F_2$ with constant rows and columns can be summed. See Ex. 28 on p. 301 of Whittaker and Watson; formulas (2.6.7)–(2.6.14) of Andrews/Askey/Roy.

For a good list of special hypergeometric values, see Appendix III of L. J. Slater, *Generalized Hypergeometric Functions*.

It is worth knowing that any ${}_2F_1$ with a top parameter $-n$ can be written as a Jacobi polynomial. (See AAR p. 99.) Jacobi polynomials are a classical family that are orthogonal with respect to beta distributions on $[-1, 1]$.

Some possibly useful references: Srivastava and Manocha, A Treatise on Generating Functions [Math QA 353 G44 S 68 1984]; Knottnerus, Approximation Formulae for Generalized Hypergeometric Functions for Large Values of the Parameter [Math QA 351 K 67]; Agarwal, Generalized Hypergeometric Series [Memorial (!) QA 351 A4].

An ${}_3F_2(1)$ can be expressed as a "log moment" of a beta distribution, using Euler's integral (Andrews/Askey/Roy p. 67).

Lecture 8

Topics du jour: Random permutations

References: Feller v. 1 p. 256 ff; PB 3.7

Why study random permutations?

These are the traditional input model for analyzing sorting and selection problems.

Properties of random permutations "transfer" to other, more complicated situations. (More about this later.)

Inversions

Let $\sigma_1, \ldots, \sigma_n$ be a permutation of $\{1, \ldots, n\}$. An *inversion* is a pair (i, j) with $i < j$ and $\sigma_i > \sigma_j$.

Intuitively, an inversion is a pair that is out of order.

Example: Consider

$i =$	1	2	3	4	5
$\sigma_i =$	3	1	4	5	2

The inversions are $(1,2), (1,5), (3,5), (4,5)$.

We can read the inversions off a "crossing diagram" such as the following:

1 2 3 4 5

3 1 4 5 2

(Draw lines linking the two copies of each number.) For example the first crossing involves the edges above 3 and 1. These are at positions 1 and 2, and we note that $\sigma_1 = 3 > \sigma_2 = 1$. Thus $(1,2)$ is an inversion.

We can also associate the first crossing with the inversion $(1,3)$ of σ^{-1}. Continuing with this kind of reasoning, we can prove that σ and σ^{-1} have the same number of inversions. (Exercise.)

To study the inversions in a random permutation, we imagine building a random permutation in the following way:

Start with an empty list.

For $i = 1, \ldots, n$: insert i into one of the i slots defined by the list so far built.

We will hold i responsible for all the inversions created by its insertion. For example, if we take the ordering

1324

and add 5 to obtain

15324

then this creates three new inversions.

Let X_i be the number of inversions created by inserting i. Evidently X_i is distributed uniformly in the set $\{0, \ldots, i-1\}$.

From our work on the discrete uniform distribution we obtain

$$E[X_i] = \frac{i+1}{2} - 1 = \frac{i-1}{2}.$$

and

$$\sigma^2[X_i] = \frac{i^2-1}{12}.$$

Let I be the total number of inversions. We have

$$E[I] = \sum_{i<j} \Pr[\ (i,j) \text{ is an inversion }] = \binom{n}{2}\frac{1}{2} = \frac{n(n-1)}{4} \sim \frac{n^2}{4}.$$

Since I is the sum of the independent random variables X_i, we have

$$\sigma^2(I) = \sum_{i=1}^{n} \sigma^2(X_i) = \sum_{i=1}^{n} \frac{i^2-1}{12} = \frac{2n^3+3n-54}{72} \sim \frac{n^3}{36}.$$

Asymptotically the number of inversions is normally distributed. For this you need to use a version of the central limit theorem for non-identically distributed random variables. This is not part of our course, but we can at least tell you what the result is. For fixed α,

$$\Pr\left[\frac{\#\text{ of inversions} - n^2/4}{n^{3/2}/6} \le \alpha\right] \sim \frac{1}{2\pi}\int_{-\infty}^{\alpha} e^{-t^2/2}dt.$$

as $n \to \infty$.

Application to insertion sorting.

Here is a tried and true method for sorting n keys $A_1, \ldots, A_n$. For $j = 2, \ldots, n$, insert A_j into the already sorted list formed by the entries $A_1, \ldots, A_{j-1}$.

We will assume that we try to find a place for A_j by starting at the right hand end of the list and locating the maximum k for which $A_k < A_j$. (If no such k exists, we stick A_j at the left of the list.)

The easiest variant of this algorithm to analyze is the "sentinel" version which has the extra element $A_0 = -\infty$. In this case, the number of key comparions to insert A_i is 1 plus the number of $j < i$ for which $A_j < A_i$. Thus

$$[\#\text{ of key comparisons}] = n + [\#\text{ of inversions in the input ordering}].$$

Let's assume the input is randomly ordered. For the total work W we have

$$E[W] \sim \frac{n^2}{4}$$

with a standard deviation of

$$\sigma[W] \sim \frac{n^{3/2}}{6}$$

Since the standard deviation is small compared to the mean, we can assert that with high probability, $\Omega(n^2)$ work is required.

This can be made precise using Chebyshev's inequality, which states that

$$\Pr[|X - \mu| \geq t\sigma] \leq 1/t^2.$$

(This is one of the few really general inequalities in probability. It is applicable whenever the first two moments of X exist.)

Using Chebyshev's inequality we find

$$\Pr[W \leq n^2/8] \sim \Pr[|W - n^2/4| \geq (3/4)\sqrt{n}\sigma] \leq \frac{2 + o(1)}{n}.$$

which goes to 0 with n.

Cycles in permutations

We now view a permutation as a mapping f from $\{1, \ldots, n\}$ to itself.

A *cycle* is a sequence $(x_1 x_2 \ldots x_\ell)$ with $f(x_i) = x_{i+1}$. (Indices taken mod ℓ.)

Every permutation may be factored into disjoint cycles, in a way that is unique except for ordering.

Example: Suppose we have

$$\begin{array}{ccc} 1 & & 2 \\ 2 & & 4 \\ 3 & \mapsto & 1 \\ 4 & & 3 \\ 5 & & 5 \end{array}$$

The cycles are (1243) and (5). So our permutation

$$\sigma = (2413)(5).$$

We can understand the cycles in random permutations by using the following procedure (due to Feller) for generating a random permutation.

Imagine that you have a shuffled deck of cards labelled 1 through n. Deal out cards until 1 appears. This makes the first cycle. For example, if the cards come out 5, 2, 3, 1 then the first cycle is

$$(1523)$$

Now repeat this process recursively (taking the "target" to be the smallest undealt card) until all cycles are made.

Analysis: if there are k cards left in the pack exactly one of them will make a cycle. Hence the probablility that this card makes a cycle is $1/k$. This event is also independent of anything that happens before or after.

Hence, the total number of cycles is the sum of n independent 0/1 random variables, with expectations

$$1/n, 1/(n-1), \ldots, 1/2, 1.$$

We have seen this random variable before: it is the number of record values in a random permutation of n distinct real numbers. From this we get as in the first lecture

$E[\#\text{ of cycles}] = H_n \sim \log n.$

$\sigma^2[\#\text{ of cycles}] = H_n - H_n^{(2)} \sim \log n.$

The number of cycles is asymptotically normally distributed.

A useful analogy.

Roughly speaking, the following objects behave in the same way (at least asymptotically):

Cycle lengths in random permutations

Sizes of factors of random numbers

Degrees of factors of random polynomials over a finite field

To make the analogy precise, the "sizes" have to be normalized, so we consider:

$|C|/n$, where C is a cycle in a permutation on n letters

$\log p/\log n$, where p is a prime divisor of n

$\deg f/\deg g$, where f is a (monic) irreducible divisor of g.

The advantage of this viewpoint is that it allows us to easily guess what is going on for numbers and polynomials, even if these results are very hard to prove rigorously. For example, the analog of the result that a random permutation has a normally distributed number of cycles is the Erdős-Kac theorem, which says that the number of prime factors of a random $n \leq x$ has a distribution that is asymptotically normal, with mean and variance $\sim \log\log x$.

For more on this idea (and some rigorous results), see P. Billingsley, Periodica Mathematica Hungarica, 1972; D. Knuth and L. Trabb Pardo, Theoretical Computer Science, 1976;

More references

R. Arratia, A. D. Barbour, S. Tavaré, Notices of the AMS, September 1997. Nice survey article on the connections between prime factorizations and random combinatorial structures.

E. Bach, SIAM J. Computing, 1988. Algorithm (based on the random bisection idea) for generating random numbers in factored form.

E. Bach, Proc. 1994 ACM-SIAM Symposium on Discrete Algorithms. Analysis of an algorithm for generating the "gaps" between random bisection points in decreasing order.

Lecture 9

Topics du jour: Stirling numbers, random bisection.

References: PB 3.7

What are Stirling numbers and why do we need them?

These are counting numbers, analogous to binomial coefficients, that occur in various combinatorial applications. For example, the exact probabilities for occupancy involve Stirling numbers.

In addition, they give a recipe for changing back and forth between powers X^n and falling powers $X^{\underline{n}}$. This is useful, as you might guess, in moment calculations.

Stirling numbers of the first kind

We define

$$\begin{bmatrix} n \\ k \end{bmatrix} = \text{ \# of permutations on } n \text{ letters with } k \text{ cycles .}$$

By the results of the last lecture, this is also the number of permutations of $\{1, \ldots, n\}$ that have k "record" values.

Properties

1. $$\begin{bmatrix} n \\ n \end{bmatrix} = 1$$

   (Only the identity permutation has n cycles.)

2. $$\begin{bmatrix} n \\ n-1 \end{bmatrix} = \binom{n}{2}$$

   (Except for one two-cycle, all elements are fixed. Any subset of size 2 leads to a different 2-cycle.)

3. $$\begin{bmatrix} n \\ 1 \end{bmatrix} = (n-1)!$$

   (A n-cycle must have the form $(1 * \ldots *)$.)

4. [Recurrence relation.]

   $$\begin{bmatrix} n \\ i \end{bmatrix} = (n-1) \begin{bmatrix} n-1 \\ i \end{bmatrix} + \begin{bmatrix} n-1 \\ i-1 \end{bmatrix}.$$

   (To build a permutation of $1, \ldots, n$ with i cycles, we can: a) form a permutation of $\{1, \ldots, n-1\}$ with i-cycles, and put n into one of these cycles, following one of the

$n-1$ items, or b) form a permutation of $\{1,\ldots,n-1\}$ with $i-1$ cycles, then put n into a cycle by itself.)

5. By convention we define $\begin{bmatrix}0\\0\end{bmatrix} = 1$. For $k<0$ and $k>n$, we have
$$\begin{bmatrix}n\\k\end{bmatrix} = 0.$$

6. We have
$$X^{\underline{n}} = \sum_i (-)^{n-i} \begin{bmatrix}n\\i\end{bmatrix} X^i.$$
I don't know a combinatorial interpretation for this. (Maybe inclusion-exclusion?) It can be proved by induction on n. The cases $n=0,1$ are immediate. To go from n to $n+1$, compute
$$\begin{aligned}
X^{\underline{n+1}} &= X^{\underline{n}}(X-n)\\
&= \sum_i (-)^{n-i} \begin{bmatrix}n\\i\end{bmatrix} X^i (X+n)\\
&= \sum_i (-)^{n-i} \begin{bmatrix}n\\i\end{bmatrix} X^{i+1} - \sum_i (-)^{n-i} n \begin{bmatrix}n\\i\end{bmatrix} X^i\\
&= \sum_i (-)^{n+1-i} \begin{bmatrix}n\\i-1\end{bmatrix} X^i + \sum_i (-)^{n+1-i} n \begin{bmatrix}n\\i\end{bmatrix} X^i\\
&= \sum_i (-)^{n+1-i} \begin{bmatrix}n+1\\i\end{bmatrix} X^i.
\end{aligned}$$
This tells us the change of basis matrix for going from powers of X to falling powers of X. For example
$$\begin{pmatrix}1\\X\\X^{\underline{2}}\\X^{\underline{3}}\\\vdots\end{pmatrix} = \begin{pmatrix}1&&&&\\0&1&&&\\0&-1&1&&\\0&2&-3&1&\\\vdots&\vdots&\vdots&\vdots&\vdots\end{pmatrix}\begin{pmatrix}1\\X\\X^2\\X^3\\\vdots\end{pmatrix}$$

7. We have
$$\begin{bmatrix}n\\i\end{bmatrix} = \sum_{k\geq 0} (n-1)^{\underline{k}} \begin{bmatrix}n-1-k\\i-1\end{bmatrix}.$$
To prove this, group the permutations with i cycles according to how long 1's cycle is. We get
$$\begin{aligned}
(1) + \text{an } i-1 \text{ cycle perm of } n-1 \text{ elements} &\quad\to \begin{bmatrix}n-1\\i-1\end{bmatrix}\\
(1*) + \text{an } i-1 \text{ cycle perm of } n-2 \text{ elements} &\quad\to (n-1)\begin{bmatrix}n-2\\i-1\end{bmatrix}\\
(1**) + \text{an } i-1 \text{ cycle perm of } n-3 \text{ elements} &\quad\to (n-1)(n-2)\begin{bmatrix}n-3\\i-1\end{bmatrix}
\end{aligned}$$

and so on. So the total number of i-cycle permutations is

$$\left\{ {n \atop i} \right\} = (n-1)^{\underline{0}} \left[{n-1 \atop i-1} \right] + (n-1)^{\underline{1}} \left[{n-2 \atop i-1} \right] + (n-1)^{\underline{2}} \left[{n-3 \atop i-1} \right] + \cdots .$$

Some sums involving Stirling numbers.

From the combinatorial interpretation we have

$$\sum_k \left[{n \atop k} \right] = n!$$

Also

$$\sum_k k \left[{n \atop k} \right] = n! \sum_k k \frac{\left[{n \atop k} \right]}{n!} = n! \sum_k k \Pr[\text{ perm has } k \text{ cycles }]$$
$$= n! E[\text{ \# of cycles }] = n! H_n.$$

Using the variance one can evaluate

$$\sum_k k^2 \left[{n \atop k} \right]$$

(exercise).

Stirling numbers of the second kind.

We define

$$\left\{ {n \atop k} \right\} = \text{ \# of equivalence relations on } \{1, \ldots, n\} \text{ with } k \text{ equivalence classes .}$$

Reminder: An equivalence relation is a binary relation that is reflexive, symmetric, and transitive. Examples include equality and (for numbers) congruence mod n.

The total number of equivalence relations on a universe of size n, in our notation $\sum_k \left\{ {n \atop k} \right\}$, is called the n-th *Bell number*. More about these later.

Properties

1.
$$\left\{ {n \atop n} \right\} = \left\{ {n \atop 1} \right\} = 1$$

(Only the identity relation has n classes, and only the trivial relation – everything equivalent to everything else – has one class.)

2.
$$\left\{ {n \atop n-1} \right\} = \binom{n}{2}$$

(Choose two elements to be identified. All others are in a class by themselves.)

3.
$$\left\{ {n \atop k} \right\} \leq \left[{n \atop k} \right]$$

(Arrange $1, \ldots, n$ into k groups, e.g. (13)(4)(256). There will be more possibilities, or at least no fewer, if we distinguish different cyclic orderings within the groups.)

4. [Recurrence relation.]
$$\left\{ {n \atop i} \right\} = i \left\{ {n-1 \atop i} \right\} + \left\{ {n-1 \atop i-1} \right\}.$$

(Similar to the proof of the recurrence relation for Stirling numbers of the first kind. Either put n in a class by itself, or add it to one of the i classes of an equivalence relation on $n-1$ elements.)

5. Convention: $\left\{ {0 \atop 0} \right\} = 1$. Theorem:
$$\left\{ {n \atop k} \right\} = 0$$

whenever $k < 0$ or $k > n$.

6. We have
$$X^n = \sum_i \left\{ {n \atop i} \right\} X^{\underline{i}}.$$

This can be proved by induction on n (GKP p. 248), or by combinatorial reasoning (below).

Application to occupancy.

We throw k balls into N bins. The random variable X denotes the number of occupied bins.

We have
$$\Pr[X = r] = \frac{N^{\underline{r}} \left\{ {k \atop r} \right\}}{N^k}.$$

Why is this? Distinguish the balls somehow (later we can always forget that we did that). There are N^k ways to assign the balls to the bins. Now consider a particular value of r. We can divide the balls into r classes (in $\left\{ {k \atop r} \right\}$ ways), and then assign the classes to the bins (in $N(N-1)\ldots(N-r+1)$ ways). Each of these gives a unique assignment, and every assignment can be obtained in this way.

Since probabilities must sum to 1 we have

$$\sum_r r! \binom{N}{r} \left\{ {k \atop r} \right\} = N^k,$$

that is,

$$\sum_r N^{\underline{r}} \left\{ {k \atop r} \right\} = N^k.$$

Since both sides are polynomials in N and equal for $N \geq 1$, this is an identity holding for arbitrary values of N.

Notes

The k-th moment of a Poisson random variable is a degree k polynomial in its parameter λ. The coefficients are Stirling numbers of the second kind. This is because the k-th factorial moment is λ^k (David and Barton, p. 77).

Bell numbers also count the number of possible patterns (AAAA, ABAA, ABCA, etc.) in a word of n letters. The obvious bound is $B(n) \leq n^n$. This is not too far from the truth as we have asymptotically

$$\log B(n) = n \log n - n \log \log n - n + ...$$

Possible reference: Moser and Wyman, Trans. Roy. Soc. Canada Sect. III (3), v. 49, 1955, pp. 49-54.

Berend and Tassa [Prob. Math. Stat. 30:2, 2010, pp. 185-205] have some nice explicit bounds. It follows from their work that

$$B(n) \leq (n/\log(n+1))^n.$$

Stirling number notation is not standard. Older books (David and Barton, in particular) refer to "differences of zero" and write $\Delta^k 0^n$ for our $\left\{ {n \atop k} \right\}$. Abramowitz and Stegun call it $\mathbf{S}_n^{(k)}$.

Knuth I gives formulas for $\sum_k k^{\underline{\nu}} \left[{n \atop k} \right]$ (essentially a factorial moment for the cycle count).

The row maximum of $\left[{n \atop k} \right]$ is an interesting problem. Jordan (Calculus of Finite Differences) asserts that the maximum is always taken at $\lfloor \log n \rfloor + 1$, but I don't find his argument to be entirely rigorous.

Lecture 10

Topics du jour: Generating functions, Wilf's snake oil method

References: PB 3.7, GK 2.1.1. See also R.P. Stanley, Enumerative Combinatorics, Vol. 1, Chapter 1 (for formal power series); H. S. Wilf, in J. Seimons, ed., Surveys in Combinatorics, 1989 (for snake oil).

Formal power series

Definition. Let $a_0, a_1, \ldots$ be a sequence of numbers (real or complex). The expression

$$f(x) = \sum_{i=0}^{\infty} a_i x^i$$

is called a *formal power series*.

We can add, subtract, and multiply formal power series.

Let $f = \sum a_i x^i$, $g = \sum a_i x^i$. Then

$$f \pm g = \sum (a_i \pm b_i) x^i$$

and

$$fg = \sum c_i x^i, \qquad \text{where } c_i = \sum_{j+k=i} a_j b_k.$$

These operations satisfy the usual laws of algebra. (More precisely: formal power series form a commutative ring.)

Division is problematic.

The obstruction is that x has no multiplicative inverse.

Theorem. If $f = \sum_i a_i x^i$, then there is a g with $fg = 1$ iff $a_0 \neq 0$.

Differentiation

Suppose $f = \sum_i a_i x^i$. Then we define

$$f' = \sum_{i \geq 1} i a_i x^{i-1}.$$

This is purely a formal (syntactic, no semantics) operation. It obeys the usual rules of calculus, for example

$$(fg)' = f'g + fg'$$

and

$$(f^{-1})' = -f^{-2} f'.$$

Series solutions to differential equations

Roughly speaking, we can do anything we like to a formal power series as long as there is a method for determining the n-th coefficient in a finite number of steps.

Example: finding solutions to simple differential equations.

If $f = \sum_i a_i x^i$, and $f' = f$, then by matching coefficients we obtain

$$a_0 = a_1 = 2a_2 = 3a_3 = \cdots,$$

so

$$\begin{aligned} f(x) = &\ a_0 \left(\sum_{i \geq 0} \frac{x^i}{i!} \right). \\ = &\ a_0 \exp(x) \end{aligned}$$

Similarly, $f'(x) = (1-x)^{-1}$ has the solution

$$a_0 + \left(\sum_{i \geq 1} \frac{x^i}{i} \right) = a_0 + \log(1/(1-x)).$$

Convergence

Many applications of power series only require matching coefficients; for these, convergence is not an issue. But you may as well know what the situation is.

The usual way to prove a power series converges is to bound it by a geometric series.

If $|a_n| < R^{-n}$, then for $|x| < R$,

$$|f(x)| \leq \sum_i |a_i||x|^i = \sum_i |x/R|^i < \infty.$$

A more precise version of this result was proved by Hadamard. Let

$$1/R = \limsup \sqrt[n]{|a_n|}.$$

R is called the *radius of convergence*. Then $f(x)$ converges absolutely when $|x| < R$, diverges when $|x| > R$. (Convergence on the boundary is tricky and can go either way.)

Generating functions

Definition: If $\{a_i\}$ is a sequence, its *generating function* is $f(x) = \sum_i a^i x^i$.

The purpose of generating functions is to reduce questions about sequences to algebraic manipulations.

Theme (please read this in a LOUD voice): The complexity of a sequence reflects the complexity of its generating function, and vice versa.

Theorem: $f(x) = a_0 + a_1 x + a_2 x^2 + \ldots$ is a rational function (ratio of polynomials) iff its coefficients a_i satisfy a relation of the form

$$a_n + A_{d-1} a_{n-1} + \cdots + A_0 a_{n-d} = 0$$

in which $A_0, \ldots, A_{d-1}$ are constants.

This is called a *linear recurrence relation with constant coefficients.*

Proof of the theorem. Write out the relevant multiples of f:

$$\begin{aligned}
f &= a_0 + a_1 x + \cdots + a_n x^n + \cdots \\
A_{d-1} x f &= A_{d-1} a_0 x + \cdots + A_{d-1} a_{n-1} x^n + \cdots \\
A_{d-2} x^2 f &= \cdots + A_{d-2} a_{n-2} x^n + \cdots \\
&\cdots \\
A_0 x^d f &= \cdots + A_0 a_{n-d} x^n + \cdots
\end{aligned}$$

Summing these, we get

$$\left(1 + A_{d-1} x + \cdots + A_0 x^d\right) f$$

$$= \text{ degree } d-1 \text{ polynomial } + \sum_{n \geq d} \left[a_n + A_{d-1} a_{n-1} + \cdots A_0 a_{n-d}\right] x^n.$$

If $f = g/h$, then assume (wolog) that h has constant coefficient 1, i.e.,

$$h = 1 + A_{d-1} x + \cdots + A_0 x^d.$$

Since hf is a polynomial, the recurrence relation must hold for all sufficiently large n. Conversely, if the recurrence relation holds, the above expression shows that f is rational.

The polynomial

$$Y^d + A_{d-1} Y^{d-1} + \cdots + A_1 Y + A_0$$

is called the *characteristic polynomial* of the recurrence relation. Note that this is the *reverse* of the polynomial appearing in the denominator when f is given as a rational function.

Using GF's to solve recurrence relations

We illustrate with an example.

Define the Fibonacci numbers by $F_0 = F_1 = 1$, and $F_n = F_{n-1} + F_{n-2}$.

We get the generating function as follows. We have

$$\begin{aligned} f &= F_0 + F_1 x + \cdots + F_n x^n + \cdots \\ xf &= F_0 x + \cdots + F_{n-1} x^n + \cdots \\ x^2 f &= \cdots + F_{n-2} x^n + \cdots \end{aligned}$$

so

$$(1 - x - x^2) f = F_0 + (F_0 + F_1) x = 1.$$

Therefore

$$f(x) = \sum_{i \geq 0} F_i x^i = \frac{1}{1 - x - x^2}$$

Explicit form for F_i:

Write

$$f(x) = \frac{1}{(1 - \alpha x)(1 - \bar{\alpha} x)},$$

where $\alpha, \bar{\alpha}$ are zeroes of

$$Y^2 - Y - 1.$$

By the quadratic formula,

$$\alpha = \frac{1 + \sqrt{5}}{2} \text{ and } \bar{\alpha} = \frac{1 - \sqrt{5}}{2}.$$

We have a partial fraction expansion

$$f(x) = \frac{1}{(1 - \alpha x)(1 - \bar{\alpha} x)} = \frac{1}{\alpha - \bar{\alpha}} \left(\frac{\alpha}{1 - \alpha x} - \frac{\bar{\alpha}}{1 - \bar{\alpha} x} \right)$$

Applying $1/(1-z) = 1+z+z^2+\ldots$ to $z = \pm \alpha x$ and matching the n-th coefficient, we get

$$F_n = \frac{\alpha^{n+1} - \bar{\alpha}^{n+1}}{\sqrt{5}}.$$

This is sometimes called the *Binet form*.

Asymptotics of F_n:

Note that $\alpha = 1.618...$ whereas $|\bar{\alpha}| = 0.618... < 1$. Therefore,

$$F_n \sim \frac{1}{\sqrt{5}} \left(\frac{1 + \sqrt{5}}{2} \right)^{n+1}$$

Wilf's snake oil method for combinatorial sums

The basic idea is to write down the generating function of the (unknown) sum, interchange the order of summation, and then by doing the inner summation, obtain a closed form for the sum. The method works more or less well depending on whether you get a generating function you recognize in the last step.

Example 1. What is

$$\sum_{k=0}^{n} \binom{k}{n-k}?$$

The generating function is

$$\begin{aligned}
F(x) &= \sum_{n\geq 0}\sum_{k\geq 0} \binom{k}{n-k} x^n \\
&= \sum_{k\geq 0} x^k \sum_{n\geq 0} \binom{k}{n-k} x^{n-k} \\
&= \sum_{k\geq 0} x^k \sum_{n} \binom{k}{n-k} x^{n-k} \\
&= \sum_{k\geq 0} x^k \sum_{j} \binom{k}{j} x^{j} \\
&= \sum_{k\geq 0} x^k (1+x)^k \\
&= \sum_{k\geq 0} (x+x^2)^k \\
&= \frac{1}{1-x-x^2}.
\end{aligned}$$

Since the coefficients are the Fibonacci numbers, we have

$$\sum_{k=0}^{n} \binom{k}{n-k} = F_n.$$

Example 2. What is

$$S_{r,n} = \sum_{k=0}^{n} \binom{r+k}{k}?$$

Here we have two free variables (r and n). We choose to sum over r.

$$G(z) = \sum_{r\geq 0} S_{r,n} z^r = \sum_{k=0}^{n}\sum_{r\geq 0} \binom{r+k}{k} z^r = \sum_{k=0}^{n}\sum_{r\geq 0} \binom{r+k}{r} z^r$$

Recognizing that

$$\sum_{r\geq 0}\binom{r+k}{r}z^r = \frac{1}{(1-z)^{k+1}},$$

and letting $w = 1-z$, we get

$$G(z) = \frac{1}{w^{n+1}}(w^n + \cdots w + 1) = \frac{1}{z}\frac{1-w^{n+1}}{w^{n+1}} = \frac{1}{z}\left(\frac{1}{(1-z)^{n+1}} - 1\right).$$

Therefore

$$G(z) = \frac{1}{z}\sum_{r\geq 1}\binom{r+n}{r}z^r = \sum_{r\geq 1}\binom{r+n}{n}z^{r-1} = \sum_{r\geq 0}\binom{r+n+1}{n}z^r.$$

Matching coefficients, we find that

$$\sum_{k=0}^{n}\binom{r+k}{k} = \binom{r+n+1}{r}.$$

Notes.

The following book has a huge number of generating function formulas and looks like it merits closer attention: I. J. Schwatt, An Introduction to the Operations with Series, Univ. Pennsylvania Press, 1924. [Math LF AC .SCH 9]

Chrystal v. 2 is also good for this kind of stuff.

It is worthwhile to know the generating functions for classical orthogonal polynomials. For example, the Legendre polynomials are generated by

$$\frac{1}{\sqrt{1-2xt+t^2}} = \sum_{n\geq 0} P_n(x)t^n.$$

A good list of such generating functions appears in Abramowitz and Stegun.

Lecture 11

Topics du jour: Using the snake oil method to analyze bucket sorting.

References: Original for this course.

Bucket sorting

Given $X_1, \ldots, X_n$, numerical keys to be sorted. We'll assume (after normalizing) that $0 \le X_i < 1$.

Bucket sorting method: Divide the interval $[0, 1)$ into k *buckets*. The i-th bucket will be $B_i := [(i-1)/k, i/k)$. Group the keys into buckets and sort each bucket individually.

Probabilistic model: X_i are i.i.d. $U(0, 1)$.

Distribution of keys into buckets follows occupancy model.

Number of keys in a bucket has the Binomial(n, p) distribution, where $p = 1/k$. (We also put $q = 1 - p$.)

Goal of analysis: figure out how the expected sorting time varies with n and k.

Easy case: Use an n^2 sorting algorithm to sort buckets.

Let m be the number of keys in a bucket. These are identically distributed (but not independent).

Ignoring the leading constant (we can always put it back later),

$$E[\#\text{ of comparisons}] = kE[m^2] = k(npq + n^2p^2) \le n + n^2/k.$$

Taking $k = n$, the mean number of comparisons is $O(n)$.

Harder question: what if we use an $O(n \log n)$ sorting algorithm?

Example: quicksort uses $2nH_n + O(n)$ comparisons on average. (Here H_m is the m-th harmonic number.)

We will show (below) that

$$E[mH_m] \le n \log(n/k) + O(n + k)$$

Thus, the work for sorting ranges from $O(n \log n)$ (1 bucket) to $O(n)$ (n buckets).

Harmonic moments for binomial random variables.

Let m have the Binomial(n, p) distribution.

We first use the snake oil method to compute

$$E_0(n) = \sum_{m\geq 0} H_m \binom{n}{m} p^m q^{n-m}.$$

The generating function is

$$\begin{aligned} f(z) = \sum_{n\geq 0} E_n z^n &= \sum_{m\geq 0} H_m p^m q^{-m} \left(\sum_{n\geq m} \binom{n}{m} (qz)^n \right) \\ &= \sum_{m\geq 0} H_m p^m q^{-m} \left(\sum_{i\geq 0} \binom{m+i}{m} (qz)^{m+i} \right) \\ &= \sum_{m\geq 0} H_m \frac{(pz)^m}{(1-qx)^{m+1}} \end{aligned}$$

Let us set $R = (pz)/(1-qz)$. Then

$$\begin{aligned} f(z) &= \frac{1}{1-qz} \sum_{m\geq 0} H_m R^m \\ &= \frac{1}{1-qz} \sum_{m\geq 0} \frac{1}{1-R} \log((1-R)^{-1}) \\ &= \frac{1}{1-z} \log\left(\frac{1-qz}{1-z}\right) \\ &= \frac{1}{1-z} \log\left(\frac{1}{1-z}\right) - \frac{1}{1-z} \log\left(\frac{1}{1-qz}\right). \end{aligned}$$

In general, if g is the generating function for a_n, then $g/(1-z)$ is the generating function for $A_n = \sum_{i\leq n} a_i$. Using this idea, we get

$$\frac{1}{1-z} \log\left(\frac{1}{1-z}\right) = \sum_{n\geq 0} H_n z^n$$

and

$$\frac{1}{1-z} \log\left(\frac{1}{1-qz}\right) = \sum_{n\geq 0} H_n(q) z^n,$$

where

$$H_n(q) = q + q^2/2 + \ldots + q^n/n$$

This implies that

$$E_0(n) = H_n - H_n(q).$$

(Compare Knuth, vol. 1 p. 76.)
In our analysis of sorting, we will need

$$E_1(n) = E[mH_m].$$

We have

$$\begin{aligned}
E_1(n) &= \sum_{m\geq 1} mH_m \binom{n}{m} p^m q^{n-m} \\
&= \sum_{m\geq 1} m(H_{m-1} + 1/m) \binom{n}{m} p^m q^{n-m} \\
&= (1-q^n) + n \sum_{m\geq 1} H_{m-1} \binom{n-1}{m-1} p^m q^{n-m} \\
&= (1-q^n) + np \sum_{m\geq 1} H_{m-1} \binom{n-1}{m-1} p^{m-1} q^{n-1-(m-1)} \\
&= (1-q^n) + npE_0(n-1).
\end{aligned}$$

Therefore,

$$E[mH_m] = np[H_{n-1} - H_{n-1}(q)] + (1-q^n).$$

Using similar ideas, we can compute

$$E_k(n) = E[m^{\underline{k}} H_m].$$

(Give recurrence here?)

Back to sorting.

Suppose we use Quicksort to sort the buckets. We will need an accurate estimate for the average number of comparisons done by this algorithm.

We will need an accurate estimate on the average number of comparisons done by Quicksort. It's known that the expected number of comparisons is

$$2(m+1)H_m - 4m \leq 2mH_m.$$

(We will prove this result a couple of lectures from now, but if you are impatient you can find it in Rawlins, Compared to What?, p. 257.)

We also need a lower bound:

$$\begin{aligned}
H_{n-1}(q) &= \log\left(\frac{1}{1-q}\right) - \sum_{i\geq n} \frac{q^i}{i} \\
&\geq -\log p - \frac{q^n}{n}(1 + q + q^2 + \cdots) \\
&= -\log p - \frac{q^n}{np}.
\end{aligned}$$

From this, the expected number of key comparisons for bucket sorting is at most

$$\begin{aligned} kE[mH_m] &= k[np(H_{n-1} - H_{n-1}(q)) + (1-q^n)] \\ &\le k[(n/k)(H_{n-1} - \log k) + 1] \\ &= n(H_{n-1} - \log k) + k \\ &= n(\log(n/k) + \gamma + O(1/n)) + k \\ &= n\log(n/k) + \gamma n + k + O(1). \end{aligned}$$

Bernstein's theorem

If the number of buckets is fixed it is easy to get the leading behavior of the number of comparisons. This follows from a general theorem that can be used to estimate binomial sums.

Theorem (Bernstein). Let $u : [0,1] \to \mathbf{R}$ be continuous. Let

$$B_n(p) = \sum_{i=0}^{n} u(i/n)\binom{n}{i}p^i(1-p)^{n-i}$$

(Note that this is just $E[u(x/n)]$, when x has a binomial distribution.) Then we have (uniformly)

$$B_n(p) \to u(p).$$

For a proof, see Feller, v. 2., p. 222.

We can apply this to $u(t) = t\log t$. Since

$$x\log x = nu(x/n) + x\log n$$

we find

$$\begin{aligned} E[x\log x] &= nE[u(x/n)] + \log nE[x] \\ &\sim n(p\log p) + np\log n \\ &= np\log(np). \end{aligned}$$

When $p = 1/k$ this is $n/k\log(n/k)$, so the expected total work is k times this, or $n\log(n/k)$.

Notes

What's the variance? It should be possible to estimate this by using an inequality of McDiarmid (See R. M. Karp, Probabilistic analysis of the Euclidean TSP, in E. L. Lawler et al, ed., The Traveling Salesman Problem, p. 187).

The "$m\log m$" moment of a binomial finds application in computing unicity distance in cryptography. See Stephen M. Matyas, A computer oriented cryptanalytic solution for multiple subsitution enciphering systems, Dissertation, U. Iowa, 1974. (Probably in Shannon's paper, too.)

Lecture 12

Topics du jour: Recurrence relations (constant coefficient linear today).

References: PB chapter 5, GK chapter 2.

Recurrence relations in general

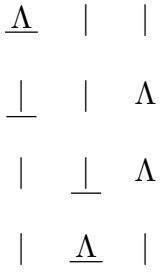
Analysis of an algorithm often shows that the running time for a given input is related to the running time for "smaller" inputs. This is especially true for recursive algorithms.

Example: A recursive solution to the Tower of Hanoi puzzle. This puzzle was described by E. Lucas around 1880.

The puzzle has 3 pegs, on the first of which are placed n disks (smaller on top).

Goal: move all disks to second peg, without placing any disk on a smaller one.

Recursive solution: move $n-1$ disks to peg 2, then largest disk to peg 3, then $n-1$ disks to peg 3.



So the number of moves satisfies

$$T_n = 2T_{n-1} + 1; \qquad T_1 = 1.$$

It is helpful to think of these relations as having the form

$$f(T_n, T_{n-1}, \ldots, T_0) = U_n, \tag{$*$}$$

where U_n is some known quantity. We will think of f as an operator on sequences, which takes a sequence $\{T_n\}$ and produces a new one $\{f(T_n)\}$.

We have two main reasons to be interested in recurrence relations.

If the relation can be solved, you get a (hopefully simple) explicit form for T_n.

Recurrence relations can also be used for computation: If you know $T_0, \ldots, T_{n-1}$, the relation tells you T_n.

The classification of recurrence relations follows that of ordinary differential equations.

Linear: the operator f in (*) is linear. If $U_n = 0$ we call the recurrence relation *homogeneous*, otherwise it is *inhomogenous*.

Non-linear: anything else.

Linear equations are important, because a) many problems naturally lead to them, and b) we can solve them! Nonlinear equations can only be handled with ad hoc tricks.

Another feature of a recurrence relation is the amount of dependence on previous values.

Linear recurrences with constant coefficients.

Assume that

$$T_n + a_{d-1}T_{n-1} + \ldots + a_0 T_{n-d} = 0,$$

where the a_i's are constants (complex numbers) and $a_0 \neq 0$. This relation is supposed to hold for sufficiently large n.

Any equation of this type can be solved using generating functions.

The idea: we know that if $\{T_n\}$ satisfies a relation of this type, then its generating function $\sum_n T_n x^n$ is rational. Using partial fractions, we can express the generating function as a polynomial plus a linear combination of terms of the form

$$\frac{1}{1-\alpha x}, \frac{1}{(1-\alpha x)^2}, \ldots, \frac{1}{(1-\alpha x)^r}.$$

with $\alpha \neq 0$.

We are primarily interested in large n, so we will ignore the polynomial.

By the binomial theorem,

$$\begin{aligned}\frac{1}{(1-\alpha x)^k} &= \sum_{n\geq 0} \binom{-k}{n} (-\alpha x)^n \\ &= \sum_{n\geq 0} \binom{n-1+k}{k-1} \alpha^n x^n.\end{aligned}$$

As a function of n, each $\binom{n-1+k}{k-1}$ is a polynomial of degree $k-1$. So this part of the generating function produces a linear combination of

$$\alpha^n, n\alpha^n, \ldots, n^{r-1}\alpha^n.$$

(Proof: from the generating function we get a linear combination of

$$\alpha^n, (n+1)\alpha^n, \ldots, \underline{(n+r-1)^{r-1}}\alpha^n.$$

But $\{1, x, \underline{x^2}, \ldots, \underline{x^{r-1}}\}$ has the same span as $\{1, x, x^2, \ldots, x^{r-1}\}$.)

Assembling these sequences for each α we get an explicit solution to the recurrence relation.

In practice, we can take a shortcut. The idea is that the recurrence relation restricts α and r to a finite number of values, which can be determined by factoring a polyomial associated with the recurrence relation.

From the underlying theory we know that the generating function of the sequence will be

$$\frac{f(x)}{g(x)} = p(x) + \sum_{\alpha \neq 0} \sum_{i=0}^{r_\alpha - 1} \frac{c_{\alpha,i}}{(1-\alpha x)^i}$$

where f, g, p are polynomials.

Therefore, there are nonzero constants C, C' for which

$$g(x) = C \prod_\alpha (1-\alpha x)^{r_\alpha} = C' \prod_\alpha (\alpha^{-1} - x)^{r_\alpha}$$

This tells us that α^{-1} is a zero of multiplicity r_α of g, or, what is the same thing, a zero of multiplicity r_α of the reversal of g. But the reversal of g just the characteristic polynomial

$$F(Y) = Y^d + a_{d-1} Y^{d-1} + \ldots + a_0.$$

Therefore, to solve a linear constant-coefficient recurrence relation, we do the following. First form the characteristic polynomial from the recurrence relation, and factor it completely:

$$f(Y) = \prod_\alpha (Y - \alpha)^{r_\alpha}.$$

Let

$$S = \bigcup_\alpha \bigcup_{i=0}^{r_\alpha - 1} \{i\alpha^i\}.$$

Find a linear combination of the sequences in S satisfying the boundary conditions.

Example: The gambler's ruin problem

Pete Rose and Paul Hornung play the following game. A fair coin is flipped. If it comes up heads, Pete pays Paul 1 dollar, and if it comes up tails, Pete takes 1 dollar from Paul.

Using recurrence relations, we can answer the following question: if Pete starts with A dollars and Paul with B dollars, what is the chance that Pete wins?

We can model the game as a random walk on a certain line graph. This graph has vertices $\{-A, \ldots, 0, \ldots, B\}$ and edges between vertices whose indices differ by 1. Think of vertex x as representing the state where Pete's net gain is x. The game starts out at state 0 and ends when the state becomes $-A$ or B.

Let $p(x)$ denote the probability that Pete eventually wins, starting from the state x.

Suppose the state is x, with $-A < x < B$. Conditioning on the flip of the next coin, we have

$$\Pr[\text{ Pete wins }] = \Pr[\text{ Pete wins } | \text{ heads }]\Pr[\text{ heads }] + \Pr[\text{ Pete wins } | \text{ tails }]\Pr[\text{ tails }],$$

that is,

$$p(x) = p(x+1)/2 + p(x-1)/2,$$

which we can rewrite as

$$p(x+1) - 2p(x) + p(x-1) = 0.$$

Solving the recurrence relation.

The characteristic equation for this recurrence relation is

$$X^2 - 2X + 1 = (X-1)^2 = 0.$$

This has a double root at 1.

Our general theory tells us that

$$p(x) = \alpha 1^x + \beta x 1^x = \alpha + \beta x$$

(at least when $x \geq 0$!).

The boundary conditions are $p(-A) = 0$, $p(B) = 1$, so

$$p(x) = \frac{A+x}{A+B}.$$

(You can check that this also works for negative x.)

The chance that Pete wins is $A/(A+B)$. Think of this as the worth of his initial fortune, relative to the total amount of money.

This is an interesting connection with electrical theory which I leave to you to explore. Suppose we connect a 1-volt battery to the end points of our graph, so that B has a potential of 1 volt and A has a potential of 0. Assume each edge has the same (positive) resistance. Then the voltage at x is the probability that Pete wins starting from x.

Example: Ladder network resistance

This gives an example of a nonlinear recurrence relation that we can put into linear form.

Consider a ladder with $k+1$ rungs. For example, $k=3$ would be

$$\begin{array}{ccccccccc} A & * & - & * & - & * & - & * \\ & | & & | & & | & & | \\ B & * & - & * & - & * & - & * \end{array}$$

Each edge represents a unit resistance. What is the total resistance seen between A and B?

This problem has a repetitive structure, which suggests we should try to find a recurrence relation.

Let $a \mid\mid b = \frac{ab}{a+b}$. This is the effective resistance of an a ohm and b ohm resistance in parallel.

If R_k is the resistance of the $k+1$-rung ladder then

$$R_k = 1 \mid\mid (2 + R_{k-1}) = \frac{2 + R_{k-1}}{3 + R_{k-1}},$$

with $R_0 = 1$.

Linearizing the recurrence relation

Put $R_k = N_k/D_k$. Then we have

$$\frac{N_k}{D_k} = \frac{N_{k-1} + 2D_{k-1}}{N_{k-1} + 3D_{k-1}}.$$

Therefore, it will suffice to solve the linear recurrence

$$\begin{aligned} N_k &= N_{k-1} + 2D_{k-1} \\ D_k &= N_{k-1} + 3D_{k-1} \end{aligned}$$

with $N_0 = D_0 = 1$.

Let $X_k = \begin{pmatrix} N_k \\ D_k \end{pmatrix}$. Then our recurrence is

$$X_k = AX_{k-1}; \qquad X_0 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

where $A = \begin{pmatrix} 1 & 2 \\ 1 & 3 \end{pmatrix}$.

This gives us the explicit solution

$$X_k = A^k X_0.$$

Unfortunately, this is not as informative as in the 1-dimensional case, so we will have to do more work.

Suppose Z is an eigenvector of A, so that $AZ = \lambda Z$. Then $A^k Z = \lambda^k Z$. This suggests trying to write X_0 as a linear combination of eigenvectors. Can we do this? We note that the eigenvalues of A are

$$\lambda = 2 + \sqrt{3}, \qquad \mu = 2 - \sqrt{3}.$$

By a standard result of linear algebra, $\mathbf{R}^2$ has a basis consisting of eigenvectors of A. Indeed, the eigenvectors

$$Z_\lambda = \begin{pmatrix} 2 \\ 1 + \sqrt{3} \end{pmatrix}, \qquad Z_\mu = \begin{pmatrix} 2 \\ 1 - \sqrt{3} \end{pmatrix}$$

do the job. Therefore, there are constants c and d for which

$$X_k = \begin{pmatrix} N_k \\ D_k \end{pmatrix} = c\lambda^k Z_\lambda + d\mu^k Z_\mu = c\lambda^k \begin{pmatrix} 2 \\ 1 + \sqrt{3} \end{pmatrix} + d\mu^k \begin{pmatrix} 2 \\ 1 - \sqrt{3} \end{pmatrix}.$$

This gives

$$R_k = \frac{2c\lambda^k + 2d\mu^k}{(1+\sqrt{3})c\lambda^k + (1-\sqrt{3})d\mu^k}.$$

You can find c and d from the initial conditions. Here we'll just do the asymptotics. Note that

$$\lambda = 2 + \sqrt{3} > 1,$$

whereas

$$|\mu| = \sqrt{3} - 1 < 1.$$

We also observe that $c \neq 0$. (If it were, we would have $R_k = 2/(1-\sqrt{3}) > 1$, which is impossible.)

Dividing top and bottom by $c\lambda^k$ we get

$$R_k = \frac{2 + O(\alpha^k)}{1 + \sqrt{3} + O(\alpha^k)} = \sqrt{3} - 1 + O(\alpha^k),$$

with $\alpha = |\mu/\lambda| = .267949....$

The limiting value of the resistance is

$$R_\infty = 0.732... \ .$$

Already $R_1 = 0.75$ comes pretty close to this.

This technique comes from an article by M. Nodine, Fib. Quarterly 28, 1990, pp. 102-106.

See also T.L. Saaty, Modern Nonlinear Equations, p. 182.

Notes.

This lecture and Lecture 10 both contain material on constant-coefficient recurrences. This should be combined somehow.

The resistance problem could also be done by finding a 3 term recurrence relation, rather than dealing with the 2×2 matrices.

Another nice example is the "transfer matrix" method for solving the Ising model on a line. See B. Cipra, An Introduction to the Ising model, AMM v. 94, 1987, pp. 937-959.

Lecture 13

Topics du jour: Inhomogeneous linear recurrence relations; Duration of play

References: Feller I, XIV.3; E. Bach, Stat. Prob. Lett. 36, 1997.

Inhomogeneous linear recurrences

These are linear equations in sequence space.

Let $T = \{T_i\}_{i=1}^{\infty}$ be a sequence of complex numbers. Let L be a linear operator taking sequences to sequences, for example

$$L(T)_n = T_n + a_{d-1}T_{n-1} + \cdots + a_0 T_{n-d}.$$

There is no topology so we will asume that the n-th member of $L(T)$ can be determined from a finite number of elements of T.

Our goal is to solve $L(T) = U$, where U is some other sequence.

Theorem: If L is linear, then any two solutions to $L(T) = U$ differ by a solution to the corresponding homogeneous equation $L(T) = 0$.

Proof: Suppose $L(T) = L(T') = U$. Then $L(T - T') = L(T) - L(T') = 0$.

Standard attack on $L(T) = U$: Find *any* solution to this equation, and find all solutions to $L(S) = 0$. Then match $T + S$ to the boundary conditions for the problem.

Example 1: Duration of play

Recall the gambler's ruin problem, in which Peter and Paul have initial stakes of A and B respectively, and play for the flip of a coin worth 1 unit. How long can they expect to play?

This is a famous (hard) problem in elementary probability theory, discussed by De Moivre, Bernoulli, Laplace, and others. (I think we follow Laplace but I am not sure.)

We have to consider all possible initial states simultaneously. Suppose x is Peter's accumulated winnings. Let E_x be the expected duration of the game from this point.

For $-A < x < B$, we have (by conditioning on who wins the next toss)

$$E_x = (1/2)(1 + E_{x+1}) + (1/2)(1 + E_{x-1}),$$

which we can rewrite as

$$L(E) := E_x - 2E_{x-1} + E_{x-2} = -2.$$

The boundary conditions are $E_{-A} = E_B = 0$.

Guessing a particular solution.

Let's use the analogy that L is like the second derivative operator. The solutions to

$$\frac{d^2}{dx^2}y = \text{ const}$$

are quadratic polynomials. This suggests trying $E_x = x^2$.

Then

$$L(x^2) = x^2 - 2(x-1)^2 + (x-2)^2 = 2.$$

This gives us a particular solution $E_x = -x^2$.

Finding the kernel of L.

From our previous results we know that if

$$L(E_x) = 0,$$

then

$$E_x = \alpha x + \beta,$$

where α and β are constants.

Solving the recurrence relation

We must have

$$E_x = -x^2 + \alpha x + \beta.$$

The initial conditions give us the equations

$$\begin{aligned} B^2 &= \alpha B + \beta \\ A^2 &= -\alpha A + \beta \end{aligned}$$

which can be solved to obtain $\alpha = B - A$, $\beta = AB$.

The expected duration of the game is therefore

$$E_x = -x^2 + (B-A)x + AB.$$

Equal stakes

If $A = B = n$, $E_x = n^2 - x^2$.

This is maximized when $x = 0$, giving a mean duration of n^2.

Good news for gamblers: the expected duration of play is proportional to the *square* of the initial stake.

We also get some intuition about random walks. After time t, we should be (in some sense) about $\sqrt{t}$ away from the mean.

Example 2: Variance in the duration of play.

Let's compute the second moment of T_x, the time to finish starting from position x.

As before, we condition on the next coin flip and get

$$E(T_x)^2 = (1/2)E((1+T_{x-1})^2) + (1/2)E((1+T_{x+1})^2)$$

so

$$2E(T_x)^2 = 1 + 2(E_{x-1}+E_{x+1})E(T_{x-1})^2 + E(T_{x-1})^2.$$

Writing F_x for the second moment, we can put this in the form

$$F_x - 2F_{x-1} + F_{x-2} = -2 - 2(E_x + E_{x-2}).$$

We'll deal only with the case of equal stakes, where $A = B = n$. Since $E_x = n^2 - x^2$, the recurrence works out to be

$$F_x - 2F_{x-1} + F_{x-2} = 4x^2 - 8x + (6 - 4n^2).$$

We'll solve this by the method of *undetermined coefficients*.

We guess that there is a degree 4 polynomial as a particular solution, say

$$f(x) = ax^4 + bx^3 + cx^2.$$

From the symmetry of the problem we have $F_x = F_{-x}$, so we can infer that $b = 0$.

Our operator, applied to f, gives us

$$f(x) - 2f(x-1) + f(x-2) = 12ax^2 - 24ax + (14a + 2c),$$

so

$$\begin{aligned} 12a &= 4 \\ 14a + 2c &= 6 - 4n^2 \end{aligned}$$

(the equation coming from the x term is redundant). Solving these, we get $a = 1/3$ and $c = 2/3 - 2n^2$.

Since f is a particular solution, we know that

$$F_x = x^4/3 + (2/3 - 2n^2)x^2 + \beta.$$

(The x term must again vanish, by symmetry.) Using the boundary contitions $F_{\pm n} = 0$, we find

$$F_x = (1/3)x^4 + (2/3 - 2n^2)x^2 + (5n^4/3 - 2n^2/3).$$

A pretty formula for the variance

Since $\sigma^2 X = E(X^2) = E(X)^2$, we have

$$\sigma^2(T_x) = \frac{2}{3}(n^4 - n^2 - x^4 + x^2).$$

Where is the variance maximized? We note that

$$\frac{d}{dx}(n^4 - n^2 - x^4 + x^2) = -4x^3 + 2x$$

has three zeroes, at $x = 0$ and $x = \pm\sqrt{2}/2$. At these places, the variance takes its maximum, which is $n^4 - n^2$.

It is worth seeing why these should be equal. By symmetry, the variance starting from $+1$ equals the variance starting from -1. Both of these must equal the variance starting from 0, since the first step (away from 0) must lead to one of these situations, and does not introduce any extra variability.

Bottom line (equal stakes to start)

If T_0 is the duration of the game, we have

$$E(T_0) = n^2$$

and

$$\sigma(T_0) = \sqrt{2/3}n^2$$

Unlike in many other situations, the mean and standard deviation are comparable.

Additional info

Feller (v. 1 p. 90, p. 349) gives exact formulas (in generating function form) for $\Pr[T_x] = k$. I think these go back to Laplace.

A formula equivalent to our variance formula appears in a paper of I. J. Good, Random Walks on Groups, published around 1954.

Higher moments can be computed by more complicated recurrence relations, for which see E. Bach, Moments in the Duration of Play, Statist. Prob. Lett., vol. 36, 1997, pp. 1–7.

Lecture 14

Topics du jour: Variable coefficient recurrences (mostly first order).

References: Purdom and Brown 5.2, Flajolet and Sedgewick 2.7 (for divide and conquer), Hoare 1962 (for quicksort).

Linear recurrences (with arbitrary coefficients)

We'll consider the homogeneous linear equation

$$T_n + a_{d-1}T_{n-1} + \cdots + a_0 T_{n-d} = 0$$

as well as the corresponding inhomogenous version

$$T_n + a_{d-1}T_{n-1} + \cdots + a_0 T_{n-d} = U_n.$$

The left side is a linear operator L on sequences, so that we could write these equations compactly as

$$L(T) = \begin{cases} 0 & \text{(homogeneous)} \\ U & \text{(inhomogeneous)} \end{cases}$$

As with the constant coefficient case, any two solutions to the inhomogeneous equation must differ by a solution to the homogeneous equation.

First-order recurrences (important special case)

A recurrence relation of the form

$$T_n = a_n T_{n-1} + b_n$$

can always be reduced to a summation.

Before proving this, we make a "cultural" remark. Any first-order linear differential equation, of the form $y' = ay + b$, can be reduced to an integration problem, if we multiply by the right function. (See Section 3.03 of R. P. Agnew, Differential Equations, 1960.) Something similar works here.

Divide both sides by $a_1 a_2 \cdots a_n$. This gives

$$U_n = U_{n-1} + c_n,$$

where

$$U_i = \frac{T_i}{a_1 \cdots a_i}$$

and

$$c_i = \frac{b_i}{a_1 \cdots a_i}.$$

Plugging the equation for U into itself repeatedly gives

$$U_n = c_n + c_{n-1} + \cdots + c_1 + U_0,$$

and if we replace U_i and c_i by their definitions we get

$$T_n = \left(\prod_{i=1}^{n} a_i\right) T_0 + \sum_{i=1}^{n} \left(\prod_{j=i+1}^{n} a_j\right) b_i.$$

Note that the coefficient of T_0 is a solution to the homogeneous equation $T_n = a_n T_{n-1}$.

The second term is a particular solution to the equation, as can be seen by setting $T_0 = 0$.

Example 1: Mergesort

The merge sort algorithm works as follows. To sort n keys, divide them into two groups of size $n/2$. Sort each recursively, then (by repeatedly picking off the maximum), merge these into a sorted list.

Let's assume $n = 2^k$. The number of comparisons (worst case) must then satisfy

$$T(n) = 2T(n/2) + (n-1).$$

Writing $A(k)$ for $T(2^k)$ this is

$$\begin{aligned} A(k) &= 2A(k-1) + (2^k - 1) \\ &= a_k A(k-1) + b_k \end{aligned}$$

with $a_k = 2$ and $b_k = 2^k - 1$.

The general theory tells us that

$$A_k = 2^k A_0 + \sum_{i=1}^{k} 2^{k-i}(2^i - 1).$$

We have $A_0 = T(1) = 0$. (No comparisons needed to sort 1 key.)

The sum works out to

$$\sum_{i=1}^{k} 2^k - \sum_{i=1}^{k} 2^{k-i} = k2^k - (1 + \cdots + 2^{i-1}) = k2^k - 2^k + 1.$$

So, when n is a power of 2

$$T(n) = n \log_2 n - n + 1.$$

Example 2: Divide-and-conquer recurrences

Mergesort is an example of a design strategy that is very commonly used for algorithms. Reduce a problem of size n to α similar problems of size n/β, solve these, and combine these. If the work for recombination is $f(n)$, the total work will satisfy

$$T(n) = \alpha T(n/\beta) + f(n)$$

These are called *divide and conquer* recurrence relations.

For mergesort, we have $\alpha = \beta = 2$, and $f(n) = n - 1$.

If $n = \beta^k$, an exact solution to the recurrence relation is given by

$$T(n) = \alpha^k T(1) + \sum_{i=1}^{k} \alpha^{k-i} f(\beta^i).$$

Since $\alpha^k = \alpha^{\log_\beta n} = n^{\log_\beta \alpha}$, this can be rewritten as

$$T(n) = n^{\log_\beta \alpha} \left[T(1) + \sum_{i=1}^{k} f(\beta^i) \alpha^{-i} \right].$$

The asymptotic growth rate of T is determined by the interactions between α, β and f. In particular, in the sum, there is a competition between α and the growth rate of f. Typically, there are 3 regimes.

Let's examine a particular example: $f(n) = n$. Then the sum is

$$\sum_{i=1}^{k} (\beta/\alpha)^i = \begin{cases} \sim \frac{\beta}{\alpha - \beta}, & \text{if } \beta < \alpha; \\ k = \log_\beta n, & \text{if } \beta = \alpha; \\ \sim \left(\frac{\beta}{\alpha}\right)^k \frac{\alpha}{\beta - \alpha}, & \text{if } \beta > \alpha. \end{cases}$$

This gives

$$T(n) \sim \begin{cases} n^{\log_\beta \alpha} \left[T(1) + \frac{\beta}{\alpha - \beta} \right], & \text{if } \beta < \alpha; \\ \frac{n \log n}{\log \beta} + nT(1), & \text{if } \beta = \alpha; \\ \frac{\alpha}{\beta - \alpha} n + n^{\log_\beta \alpha} T(1), & \text{if } \beta > \alpha. \end{cases}$$

Example 3: Quicksort

The analysis of this algorithm leads to a first-order recurrence in which both coefficients depend on n.

First, the algorithm. To sort n distinct keys $A_1, \ldots, A_n$, choose a key x, called the *splitter*. Using $n - 1$ comparisons, group the remaining keys into those less than the splitter and those larger than the splitter. Sort these two groups of keys recursively.

The work (number of comparisons) is determined by the sizes of the various subproblems P obtained.

We can write

$$\text{total work } = n - 1 + \sum_P (\text{size}(P) - 1)\,.$$

How do the subproblem sizes behave? Let's consider a simplified continuous model, called *random splitting*.

We start with the interval $(0, n)$. At the beginning of stage i, this has been subdivided into 2^{i-1} random subintervals. Choose a random point (uniformly distributed) in each of these subintervals, producing 2^i new intervals.

This process produces a tree of intervals. Let the random variables $X_0, X_1, X_2, \ldots$ represent the lengths of the intervals in some (infinite) path from the root in the tree. Evidently we have

$$E[X_0] = n, \qquad E[X_1] = n/2,$$

and in general

$$E[X_i] = E[E[X_i|X_{i-1}]] = E[X_{i-1}/2] = n/2^i.$$

We pause to note that the random variables $Y_i = 2^i X_i$ form a *martingale* in the sense that $E[Y_i|Y_{i-1}] = Y_{i-1}$.

The subproblem sizes in quicksort are overestimated by random splitting, since the subproblems are uniform *integers* in $\{0, \ldots, n-1\}$. Therefore, if $n - i$ is the size of a subproblem at the i-th level

$$E[n_i] \le n/2^i.$$

If every subproblem has a size equal to its mean value, the number of comparisons used is asymptotic to $n \log_2 n$. Some people use this as a heuristic argument that quicksort uses $O(n \log n)$ comparisons, but as we will see, it doesn't get the constant factor right.

A correct calculation follows from conditioning on the sizes of the subproblems.

Let E_n be the expected number of comparisons used on (randomly ordered) inputs of size n.

Note that the first two subproblems generated have sizes uniformly distributed in $\{0, 1, \ldots, n-1\}$. (The sizes are not independent, since they sum to $n - 1$.)

Therefore

$$E_n = (n-1) + 2\sum_{i=0}^{n-1} E_i/n,$$

which implies

$$nE_n = n(n-1) + 2\sum_{i=0}^{n-1} E_i.$$

To solve this, first get rid of the sum by subtracting

$$n-1E_{n-1} = (n-2)(n-1) + 2\sum_{i=0}^{n-2} E_i.$$

We get

$$nE_n - (n-1)E_{n-1} = 2(n-1) + 2E_{n-1},$$

so

$$nE_n = (n+1)E_{n-1} + 2(n-1),$$

or

$$E_n = \frac{n+1}{n}E_{n-1} + \frac{2(n-1)}{n}.$$

This is a first-order linear recurrence with $a_n = (n+1)/n$, and $b_n = 2(n-1)/n$. We have $E_0 = 0$, so the solution is

$$\begin{aligned} E_n &= \sum_{i=1}^{n} \left(\prod_{j=i+1}^{n} a_j \right) b_i \\ &= 2(n+1)\sum_{i=1}^{n} \frac{i-1}{i(i+1)} \\ &= 2(n+1)\sum_{i=1}^{n} \frac{(i+1)-2}{i(i+1)} \\ &= 2(n+1)\left[H_n - 2 + \frac{2}{n+1} \right] \\ &= 2(n+1)H_n - 4n. \end{aligned}$$

It's interesting to consider a continuous analog of the quicksort recurrence. This is

$$xy(x) = x^2 + 2\int_0^x y(t)dt.$$

After differentiation, this rearranges into $xy' - y = 2x$. The solution is $y = 2x\log x + cx$.

Notes.

See A. J. B. Ward, The solution of finite-difference equations, Math. Gazette v. 82, 1998, for a nice treatment of linear second-order variable coefficient recurrences.

The parking problem (greedy random heuristic for independent set on a line) is a nice one to discuss here.

Lecture 15

Topic du jour: General linear recurrence example: debunking 3-way quicksort.

References: GK pp. 18-20.

The algorithm

This is just like ordinary quicksort except for the choice of the splitter, x. We sample (without replacement) three random keys a, b, c and let x be the median of these three. Let the rank of x be k. After grouping the keys as

$$A_1, \ldots, A_{k-1}, A_k = x, A_{k+1}, \ldots, A_n$$

we recursively sort the two lists.

This is supposed to improve the behavior of the algorithm, since x is closer to the center of the list.

Naively, you might expect a large improvement. We will show below that the average running time goes down by about 14%.

Setting up the recurrence relation

We condition on the rank of the splitter x.

$$E[\text{ work }] = E[\text{ top level work }] + \sum_{k=1}^{n} E[\text{ remaining work } | x = A_k] \Pr[x = A_k].$$

We can sort 3 keys using an average of 8/3 comparisons (exercise). Once the splitter is determined, it must be compared to $n - 3$ other keys. Thus the expected work at the top level is

$$8/3 + (n - 3) = n - 1/3.$$

What's $\Pr[x = A_k]$? There are $\binom{n}{3}$ ways to choose 3 keys, suppose they are $w < x < y$. If x has rank k, then we can still choose w in $k - 1$ ways and y in $n - k$ ways. These choices are independent, so

$$\Pr[x = A_k] = \frac{(k-1)(n-k)}{\binom{n}{3}}.$$

This is a discrete analog of the beta distribution.

Let T_n be the expected number of comparisons to solve a problem of size n. Because the left and right subproblems have the same distribution,

$$T_n = (n - 1/2) + 2 \sum_{k=1}^{n} \frac{(k-1)(n-k)}{\binom{n}{3}} T_{k-1}.$$

We put this in the form $L(T) = U$:

$$T_n - \frac{2}{\binom{n}{3}} \sum_{k=1}^{n} (k-1)(n-k) T_{k-1} = n - 1/3, \qquad \text{for } n \geq 3,$$

with initial conditions $T_0 = T_1 = 0$, $T_2 = 1$.

A related continuous model

We can get a good idea of how T_n behaves by solving the related continuous problem

$$y - \frac{2}{x^3/6} \int_0^x t(x-t)y(t)dt = x.$$

This is equivalent to

$$x^3 y - 12 \int_0^x t(x-t)y(t)dt = x^4. \qquad (*)$$

We convert this into a differential equation, as follows. Write the integral as

$$x \int_0^x ty - \int_0^x t^2 y,$$

and compute

$$\frac{d}{dx}\left[x \int_0^x ty - \int_0^x t^2 y\right] = \int_0^x ty dt,$$

so

$$\frac{d^2}{dx^2}\left[x \int_0^x ty - \int_0^x t^2 y\right] = xy.$$

Therefore, if we differentiate both sides of (*) twice, and cancel the factor x, we will have

$$L(y) := x^2 y'' + 6xy' - 6y = 12x. \qquad (**)$$

Since this comes from a sorting problem, we expect that there should be a solution proportional to $x \log x$. Note that

$$L(x \log x) = x + 6x(\log x + 1) - 6x \log x = 7x.$$

Therefore, $y = (12/7)x \log x$ is a particular solution to (**).

To get all solutions, we can solve the homogeneous version of (**). This is a Cauchy equation, which has solutions of the form x^α. Plugging this in, we get the quadratic equation $\alpha^2 - 5\alpha - 6 = 0$, so $\alpha = 1, -6$. Thus, the solutions are x and x^{-6}. Note that the first solution grows linearly, and the second is transient (decays to 0 for large x).

Back to the discrete problem.

Recall that in solving $L(T) = U$ there are two steps:

Find all solutions to $L(T) = 0$, the associated homogeneous equation.

Find one solution to $L(T) = U$.

Solving the homogeneous equation

$L(T) = 0$ is equivalent to

$$n^3 T_n = 12 \sum_{k=1}^{n} (n-k)(k-1) T_{k-1}.$$

The convolution suggests that we should use generating functions.

Put

$$G(z) = \sum_{n \geq 1} T_n z^n.$$

Then

$$z^2 G'(z) = \sum_{n \geq 1} n T_n z^{n+1} = \sum_{n \geq 2} (n-1) T_{n-1} z^n$$

We also have

$$\frac{z}{(1-z)^2} = \sum_{n \geq 1} n z^n.$$

Multiply these together and get

$$\frac{z^3 G'(z)}{(1-z)^2} = \sum_{n \geq 3} \left(\sum_{k=2}^{n-1} (n-k)(k-1) T_{k-1} \right) z^n = \sum_{n \geq 3} \left(\sum_{k=1}^{n} (n-k)(k-1) T_{k-1} \right) z^n$$

From the definition of G, we find

$$G'''(z) = \sum_{n \geq 3} n(n-1)(n-2) z^{n-3} T_n$$

so

$$z^3 G'''(z) = \sum_{n \geq 3} n(n-1)(n-2) z^n T_n$$

By the recurrence relation (homogeneous version) we have

$$z^3 G''' = \frac{12 z^3 G'}{(1-z)^2},$$

which becomes

$$F'' = \frac{12F}{(1-z)^2}.$$

if we set $F' = G$. Calculus suggests trying solutions of the form $F = (1-z)^\alpha$. Then

$$F'' = \alpha(\alpha-1)(1-z)^{\alpha-2} = 12(1-z)^{\alpha-2},$$

which is only possible if $\alpha^2 - \alpha = 12$, that is, when $\alpha = -3, 4$.

Therefore $G = \int F$ can be

$$(1-z)^5 = 1 - 5z + \cdots + (-)^n \binom{5}{n} z^n + \cdots$$

or

$$(1-z)^{-2} = 1 + 2z + \cdots + (n+1)z^n + \cdots$$

which gives two linearly independent solutions to the homogeneous recurrence:

$$T_n = \begin{cases} n+1 \\ \\ (-)^n \binom{5}{n} \end{cases}$$

As with the continuous problem, there is one with linear growth and one that is transient.

Finding a particular solution

There is, in general, no easy way to do this. In practice you have to feed stuff into the linear operator and see what comes out, hoping that you get enough different images to match the target.

The n-th term of $L(T)$ is

$$T_n - \frac{2}{\binom{n}{3}} \sum_{k=1}^{n} (k-1)(n-k) T_{k-1}$$

Recalling that the sum comes from a probability distribution, we get

$$L(1) = 1 - \frac{2}{\binom{n}{3}} \sum_{k=1}^{n} (k-1)(n-k) = 1 - 2 = -1.$$

As a warmup for what's coming, you can show as an exercise that $L((n-1)^{\underline{t}}$ is a constant for $t = 1$, and a polynomial of degree t if $t \geq 2$.

Our differential equation had a solution proportional to $x \log x$, so let's try $T_n = nH_n$.

It's convenient to deal with all powers of n times H_n simultaneously, so we'll use $T_n = (n-1)^{\underline{t}} H_n$. This form is chosen because T_{k-1} has a falling power of $k-2$, which can be combined with the $k-1$ inside the sum.

Lemma:

$$\sum_{k=1}^{n} k^{\underline{m}} = \frac{(n+1)^{\underline{m+1}}}{m+1}.$$

(Equivalent to the "hockey stick" binomial identity.)

Another Lemma: For all $m, n \geq 0$, we have

$$\sum_{k=1}^{n} \binom{k}{m} H_k = \binom{n+1}{m+1}[H_{n+1} - \frac{1}{m+1}]$$

(This is on p. 10 of GK. You can prove it using the previous lemma and partial summation.) We will use it in the form

$$\sum_{k=1}^{n} k^{\underline{m}} H_k = \frac{(n+1)^{\underline{m+1}}}{m+1}[H_{n+1} - \frac{1}{m+1}]$$

The n-th term of $L((n-1)^{\underline{t}}H_n)$ is

$$(n-1)^{\underline{t}}H_n - \frac{12}{n^{\underline{3}}}\left[S_1 - S_2 + S_3\right],$$

with

$$\begin{aligned}
S_1 &= n\sum_{k=1}^{n-1}(k-1)^{\underline{t+1}}H_{k-1} = \frac{n^{\underline{t+3}}}{t+2}\left[H_n - \frac{1}{n} - \frac{1}{t+2}\right], \\
S_2 &= \sum_{k=1}^{n-1} k^{\underline{t+2}}H_k = \frac{n^{\underline{t+3}}}{t+3}\left[H_n - \frac{1}{t+3}\right], \\
S_3 &= \sum_{k=1}^{n-1}(k-1)^{\underline{t+1}} = \frac{n^{\underline{t+3}}}{n(t+2)}.
\end{aligned}$$

(Here, we have used the lemmas.) Note that S_3 will cancel with the middle term in S_1.

Taking $t = 0, 1$, we have

$$L(H_n) = -H_n + 5/3$$

and

$$L((n-1)H_n) = 2H_n + \frac{7}{12}(n-3).$$

Therefore,

$$L(2H_n + (n-1)H_n) = L((n+1)H_n) = \frac{7}{12}n + \frac{19}{12},$$

and (using $L(1) = -1$)

$$L((12/7)(n+1)H_n + 64/21) = n - 1/3.$$

Therefore, a particular solution to the recurrence relation is

$$\frac{12}{7}(n+1)H_n + 64/21.$$

Bottom line.

We get

$$T_n = \frac{12}{7}(n+1)H_n + \frac{64}{21} + \alpha(n+1) + (-1)^n\beta\binom{5}{n},$$

with α and β determined by initial conditions. From $T_1 = 0$ and $T_2 = 1$ we get

$$\alpha = -159/49, \qquad \beta = -2/735.$$

Evaluating this formula for some small values of n, we find the values of $T_1, T_2, T_3, \ldots$ to be

$$0,\ 1,\ 8/3,\ 14/3,\ 106/15, \ldots\ .$$

The value for $n = 4$ can be checked as follows. In expectation, we use $8/3$ comparisons to sort the first 3 keys. One more comparison is needed to place the remaining key before or after the median. After this, there are two keys on one side of the median whose order must be determined. So we use, on average,

$$\frac{8}{3} + 1 + 1 = \frac{14}{3}$$

comparisons.

Lecture 16

Topic du jour: Nonlinear (polynomial) recurrence relations

References: Aho and Sloane, Fibonacci Quarterly, 1973; GK ????. The underlying theory is explained in Silverman's book, The Arithmetic of Dynamical Systems (2007). Slides from a talk on this subject were posted online.

So far, the recurrences we have studied are linear. In this lecture, we consider non-linear recurrences, given by polynomial (or rational function) iterations. The main technical tool is a concept from diophantine geometry called the height.

For such iterations, the asymptotics of a first-order recurrence in one variable

$$x_n = f(x_{n-1})$$

are determined by the degree of f and the "complexity" (log height) of initial value.

Applies to many sequences analyzed in the literature by ad hoc methods.

Once a few concepts (projective space, heights) are digested, the proofs are elementary.

Extends in some cases to handle higher-order nonlinear recurrence relations.

What are height functions?

Consider an equation like

$$x^2 + y^2 = z^2$$

with infinitely many integer solutions.

How do we measure their "complexity"?

Norms aren't useful, since (for example) $||(x, y)|| = 1$.

Paradigm: logarithms on $\mathbf{Z_{>0}}$

Naive logarithm:

$$\lg n = \text{ length of } n \text{ in binary}$$

n	binary	$\lg n$
1	1	1
2	10	2
3	11	2
4	100	3
5	...	3

and in general,

$$\lg n = \lfloor \log_2 n \rfloor + 1.$$

It is better to satisfy transformation rules

$$\log(mn) = \log m + \log n$$

without losing (much) information:

$$\log n = \lg n + O(1)$$

Néron-Tate height (1950's)

A "logarithm" for elliptic curves

Compatible with addition on the curve in the same way the logarithm is compatible with multiplication.

Projective space

This consists of all generalized ratios

$$(x_0 : x_1 : \cdots : x_n)$$

in which not all x_i are zero.

Contains ordinary n-space

$$(x_1, \ldots, x_n) \to (1 : x_1 : \cdots : x_n)$$

Rational points

Can use integer coordinates

$$(2 : 3 : 5 : 9)$$

without common factors

Naive height function

If $x_i \in \mathbf{Z}$

$$H(x_0 : \cdots : x_n) = \max |x_i|$$

We will use logarithmic form

$$h(P) = \log H(P)$$

Two useful theorems

Maps from projective n-space to m-space

We restrict to the form

$$f(x_0 : x_1 : \cdots : x_n) \quad = \quad (f_0 : f_1 : \cdots : f_m)$$

with f_i homogeneous, same degree d, no common zeroes save $(0, \ldots, 0)$

Then

$$h(f(P)) = dh(P) + O(1).$$

This says that the obvious bound is accurate, in the sense that there is never systematic cancellation.

Canonical height (Tate):

If $d > 1$ then

$$\hat{h}(x) = \lim_{n \to \infty} \frac{h(f^n(x))}{d^n}$$

Properties

$$\hat{h}(f(x)) = d\hat{h}(x)$$
$$\hat{h}(x) = h(x) + O(1)$$

analogous to log vs. lg.

Example 1

Newton iteration for $\sqrt{2}$

$$x \mapsto \frac{1}{2}\left(x + \frac{2}{x}\right)$$

$$x_0 = 2$$
$$x_1 = \frac{3}{2}$$
$$x_2 = \frac{17}{12}$$
$$x_3 = \frac{577}{408}$$
$$x_4 = \frac{665857}{470832}$$
$$x_5 = \frac{886731088897}{627013566048}$$
$$x_6 = \frac{1572584048032918633353217}{1111984844349868137938112}$$

Extends to a degree 2 map

$$(x : z) \to (x^2 + 2z^2 : 2xz)$$

so height ($\approx$ length) doubles

Example 2

Doubly exponential sequences (Aho-Sloane)

The sequence

$$x_n = 1 + x_{n-1}^2, \qquad x_0 = 1$$

counts planar embeddings of height $\leq n$ binary trees.

Any 1-variable polynomial map extends to projective space. This one gives

$$f(x : z) \mapsto (x^2 + z^2 : z^2)$$

Apply induction

$$f^n(x_0 : 1) = (x_n : 1)$$

and Tate to get

$$\begin{aligned} \log x_n &= h(x_n) + O(1) \\ &= 2^n h(x_0) + O(1) \\ &\sim \alpha 2^n \end{aligned}$$

Iteration of $x_n = x_{n-1}^2 + 1$

What happens numerically:

i	x_i	$\log x_i / 2^i$
0	1	0
1	2	.346573
2	5	.402359
3	26	.407262
4	677	.407354
5	458330	.407354
6	210066388901	.407354
7	44127887745906175987802	.407354

The growth of x_n is controlled by degree (2) and canonical height of starting point ($\alpha = 0.407354...$). So

$$x_n \approx e^{\alpha 2^n} = (1.502836...)^{2^n}$$

An Efficient Algorithm for α (Aho-Sloane?):

Consider a polynomial map $f(x) = a_d x^d + \cdots + a_1 x + a_0$, with $a_d \neq 0$.

Put

$$\frac{f(x)}{x^d} = a_d(1 + \frac{a_{d-1}}{a_d x} + \cdots + \frac{a_0}{a_d x^d}) = a_d(1 + g(x)).$$

If $x_n = f(x_{n-1})$, then

$$\frac{\log x_n}{d^n} = \log x_0 + \sum_{k=1}^{n} \frac{\log x_k - d \log x_{k-1}}{d^k}$$

$$= \log x_0 + \sum_{k=1}^{n} \frac{1}{d^k} \log \frac{f(x_{k-1})}{x_{k-1}^d}.$$

So α is

$$\hat{h}(x_0) = \log x_0 + \frac{\log a_d}{d-1} + \sum_{k=1}^{\infty} \frac{\log(1 + g(x_{k-1}))}{d^k}.$$

Rapid convergence because $g(x) = O(1/x)$.

Lecture 17

Topic du jour: Asymptotic notation, Stieltjes integrals

References: GK 4.1, 4.2. For Stieltjes integrals, see Protter and Morrey, A First Course in Real Analysis, or T. M. Apostol, Mathematical Analysis.

Asymptotic notation

Here are some standard ways to compare the growth rate of functions. Let f, g be positive functions defined in some neighborhood of $+\infty$.

1. $f = O(g)$ means there is a $C > 0$ such that $f(x) \leq Cg(x)$ for all sufficiently large x.
2. $f = \Omega(g)$ means there is a $c > 0$ such that $f(x) \geq cg(x)$ for all sufficiently large x.
3. $f = \Theta(g)$ means that $f = O(g)$ and $g = O(f)$.
4. $f \sim g$ means that $\lim_{x\to\infty} f(x)/g(x) = 1$.
5. $f = o(g)$ means that $\lim_{x\to\infty} f(x)/g(x) = 0$.

Important: big-O gives a partial order, not a total order. On certain families of functions, however, it reduces to a total order.

Example: Consider all functions of the form

$$x^a(\log x)^b(\log\log x)^c.$$

We have the theorem that

$$x^a(\log x)^b(\log\log x)^c = O(x^{a'}(\log x)^{b'}(\log\log x)^{c'})$$

iff $(a, b, c) \leq (a', b', c')$ in the sense of lexicographic (dictionary) order.

For many purposes, it helps to think of the powers of $\log x$ as small perturbations to powers of x. This can be visualized in a nice way. Put all the powers of x on a real line:

We can expand any point, say x, under a microscope to expose its "neighbors" of the form $x(\log x)^b$.

Bootstrapping

Often a function is given in implicit form and we need to develop an asymptotic representation for it. This can be done by repeatedly plugging the equation into itelf.

Example:

$$y(x) = \frac{x}{\log x}$$

has (for $x \geq e$) an inverse function $x(y)$. What is the growth rate of $x(y)$?

The definition of y implies that

$$x = y \log x.$$

Substitute this into itself to obtain

$$x = y \log(y \log x) = y \log y + y \log \log x.$$

We now want to show that the second term is negligible compared to the first. Note that for large x we have

$$\sqrt{x} \leq y \leq x$$

so (taking logs twice)

$$\log \log x - \log 2 \leq \log \log y \leq \log \log x,$$

which implies

$$\log \log y = \log \log x + O(1).$$

Combining these we get a nice result:

$$x(y) = y \log y + y \log \log y + O(y).$$

More terms could be found in the same way.

As an exercise, you can carry this one step further and derive

$$x(y) = y\log y + y\log\log y + y\frac{\log\log y}{\log y} + O\left(\frac{y}{\log y}\right).$$

Further example: The prime counting function $\pi(x)$ is often approximated by

$$z(x) = \frac{x}{\log x - 1}.$$

(This improves the "usual" result that $\pi(x) \sim x/\log x$.) To invert this function, note that

$$z/e = \frac{x/e}{\log(x/e)},$$

so (by the work above)

$$x/e = z/e[\log(z/e) + \log\log(z/e) + O(z)].$$

Therefore,

$$x \sim z[\log z + \log\log z - 1 + O(z)].$$

Asymptotic series

There are many situations where a function can be well approximated by a finite segment of a series, even if that series does not converge.

Example: Precise versions of Stirling's formula for $n!$.

We know that

$$n! \sim \sqrt{2\pi n}\left(\frac{n}{e}\right)^n$$

For a more accurate result, we can use one or more terms of the series

$$\frac{n!}{\sqrt{2\pi n}(n/e)^n} = 1 + \frac{1}{12n} + \frac{1}{288n^2} - \frac{139}{51840n^3} + \frac{571}{2488320n^4} - \cdots$$

For $n = 5$, the first five approximations given by the above series are

$$120, \quad 118.2, \quad 119.99, \quad 120.0026, \quad 120.00001, \quad 120.00006$$

It can be shown that the k-th term is C_k/n^{k-1}, with

$$C_k \sim \frac{(4k)!}{(2\pi)^k k^2}.$$

This grows rapidly, so the series does not coverge.

A definition suitable for exmples of this type was given by H. Poincaré in 1886. We say that

$$F(x) \sim a_0 + a_1/x + a_2/x^2 + \cdots + a_n/x^n + \cdots$$

is an *asymptotic series* for F if for every $n \geq 0$,

$$\left| F(x) - \sum_{i=0}^{n} a_i/x^i \right| = o(x^{-n}).$$

More briefly, the error in the truncated series is small compared to the last term used.

You should think of this as equivalent to a whole bunch of o-estimates.

There is nothing holy about the powers of x. We could use some other "scale," like x times powers of $\log x$, if we wanted.

Stieltjes integrals

Recall that the ordinary (Riemann) integral $\int_a^b f(x)dx$ is defined as the limiting value of sums of the form

$$\sum_i f(\xi_i)(x_i - x_{i-1}),$$

where $a = x_0 < x_1 < \cdots < x_n = b$ of $[a, b]$, and $x_{i-1} \leq \xi_i \leq x_i$, $i = 1, \ldots, n$.

Precisely, the integral has the value I if for each $\epsilon > 0$, there is a $\delta > 0$ such that $|\text{ sum } - I| < \epsilon$ whenever $\max\{x_i - x_{i-1}\} < \delta$.

The *Stieltjes integral* $\int_a^b f dg$ is the limiting value of sums of the form

$$\sum_i f(\xi_i)(g(x_i) - g(x_{i-1})),$$

with x_i, ξ_i as before.

As before, we give the integral the value I if for each $\epsilon > 0$, there is a $\delta > 0$ such that $|\text{ sum } - I| < \epsilon$ whenever $\max\{x_i - x_{i-1}\} < \delta$.

This is properly called a *Riemann-Stieltjes* integral.

The function g is called the *generating function*. In many applications it is non-decreasing.

It's useful (but not 100% precise) to think of this as the integral of f with respect to a "mass" or "measure" specified by g.

Evaluation can usually be reduced to one of the following two cases:

1. If g has a continuous derivative, then

$$\int_a^b f dg = \int_a^b f(t)g'(t)dt.$$

(We think of g' as a mass density in this case.)

2. Suppose that $0 \in (a, b)$ and f is continuous at 0. If g is the unit step function (0 for $t < 0$, 1 for $t \geq 0$), then
$$\int_a^b f dg = f(0).$$
(We think of a unit mass at 0. If you believe in the Dirac delta function, you can write this integral as $\int_a^b f(t)\delta(t)dt$. Observe that the Stieltjes integral allows you to make sense of this.)

Properties:

Linearity: The integral
$$\int_a^b f dg$$
is a bilinear function of f and g.

If $|f| \leq M$ and g is non-decreasing then
$$\left| \int_a^b f dg \right| \leq M(g(b) - g(a)).$$

Integration by parts: We have
$$\int_a^b f dg = [fg]_a^b - \int_a^b g df,$$
whenever either integral exists. (Included in the theorem is that existence of one integral guarantees existence of the other.)

The integral always exists when f is continuous and g is nondecreasing. However, one runs into problems when f and g share points of discontinuity. For example, $\int_{1/2}^{3/2} \lceil x \rceil d\lfloor x \rfloor$ does not exist.

Notes

Proofs of the Stieltjes integral properties can be found in Section 12.2 of Protter and Morrey, A First Course in Real Analysis.

Some authors define the Stieltjes integral using the concept of partition refinements. (Greene and Knuth do this.) This encompasses the definition above, but allows more functions to be integrated. For example, $\int_{1/2}^{3/2} \lceil x \rceil d\lfloor x \rfloor$ is now assigned the value 0. This improvement was introduced by Pollard [Quart. J. Pure. Appl. Math, 49, 1920].

There is a "Lebesgue" version of the Stieltjes integral as well. A thorough treatment of this (albeit with unfortunate choice of notation) is in Chapter XII of L.M. Graves, The Theory of Functions of Real Variables, McGraw-Hill, 1956. See also E. Hewitt, Integration by Parts for Stieltjes Integrals, American Math. Monthly v. 67, 1960, 419-423.

Lecture 18

Note: Needs to be split into two lectures.

Topic du jour: Applications of Stieltjes integrals: prime number sums, summation by parts, Euler-Maclaurin summation.

References: Landau, Primzahlen (for prime number sums); PB 2.5 (for summation by parts); GK 4.2.2 and PB 4.5.1-2 (for Euler-Maclaurin).

Review of Stieltjes integrals

$\int_a^b f dg$ is the limit of $\sum f(\theta_i)(g(x_{i+1}) - g(x_i))$

We can usually think of this as the integral of f with respect to a "measure" or "density" specified by g. Two important cases:

If $a < t < b$ and

$$g(t) = \begin{cases} 0, & x < t; \\ 1, & x \geq t. \end{cases}$$

then g represents a unit mass at t. Assuming that f is continuous at t, we get

$$\int_a^b f dg = f(t).$$

If g is C^1 (has a continuous derivative) on $[a, b]$, then

$$\int_a^b f dg = \int_a^b f(t)g'(t)dt.$$

Integration by parts: If either integral exists, we have

$$\int_a^b f dg = [fg]_a^b - \int_a^b g df.$$

As we will see, this formula is the workhorse of asymptotic approximation.

Example 1: Residue Arithmetic

Suppose $n = p_1 p_2 \dots p_r$ is a product of distinct primes. We can represent any m in the range $0 \leq m < n$ uniquely as a vector of remainders

$$(m \bmod p_1, m \bmod p_2, \dots, m \bmod p_r).$$

This representation can be used as a way to get fast, carry-free arithmetic.

Suppose we are willing to use primes $\le x$. How many numbers can we represent using this scheme? Since the residues mod p_i fit into $\log_2 p_i$ bits (more or less), we need to know the value of

$$\sum_{p \le x} \log_2 p = \frac{1}{\log_2 p} \sum_{p \le x} \log p$$

The main tool in evaluating such sums is the *prime number theorem*, which says that $\pi(x)$, the number of primes $\le x$, satisfies

$$\pi(x) = \frac{x}{\log x} + O\left(\frac{x}{(\log x)^2}\right).$$

This is a deep and famous result, which we will not prove here. (See, e.g., A. E. Ingham, *The Distribution of Prime Numbers.*)

Using the Stieltjes integral, we have

$$\begin{aligned}
\sum_{p \le x} \log p = & \int_{3/2}^{x} \log t d\pi(t) \\
= & \left[\log t \pi(t)\right]_{3/2}^{x} - \int_{3/2}^{x} \frac{\pi(t) dt}{t} \\
= & \log x (x/\log x + O(x/(\log x)^2)) - \int_{3/2}^{x} \frac{dt}{\log t} + O(\int_{3/2}^{x} \frac{dt}{(\log t)^2}) \\
= & x + O(x/(\log x)) + O(\int_{3/2}^{x} \frac{dt}{\log t}).
\end{aligned}$$

We have

$$\int_{3/2}^{x} \frac{dt}{\log t} \le \frac{x}{\log(3/2)},$$

but that is not good enough. We should be able to do better, because the integrand isn't equal to its maximum value for very long. Splitting up the range of integration, we have

$$\begin{aligned}
\int_{3/2}^{x} \frac{dt}{\log t} = & \int_{3/2}^{x/\log x} \frac{dt}{\log t} + \int_{x/\log x}^{x} \frac{dt}{\log t} \\
\le & (x/\log x)\log(3/2) + \frac{x}{\log(x/\log x)} \\
= & O(x/\log x).
\end{aligned}$$

Bottom line:

$$\sum_{p \le x} \log_2 p = \frac{x}{\log 2} + O\left(\frac{x}{\log x}\right).$$

Example 2: The sieve of Eratosthenes

To make a list of the primes up to n, we can start with $2, 3, \ldots, n$ in an array. Mark 2 as prime and cross off all its multiples (they are composite). The next unmarked number is prime, and we cross off *its* multiples, and continue similarly until every number has been classified.

The number of array accesses is

$$\sum_{p \le n} \lfloor n/p \rfloor = n \sum_{p \le n} \frac{1}{p} + O(n).$$

Using integration by parts and the prime number theorem,

$$\begin{aligned}
\sum_{p \le n} \frac{1}{p} &= \int_{3/2}^{n} \frac{d\pi(t)}{t} \\
&= \frac{\pi(n)}{n} + \int_{3/2}^{n} \frac{\pi(t)}{t^2} dt \\
&= o(1) + \int_{3/2}^{n} \frac{dt}{t \log t} + O\left(\int_{3/2}^{n} \frac{dt}{t \log^2 t} \right).
\end{aligned}$$

Using the substitution $u = \log x$, the first integral becomes

$$\int_{\log 3/2}^{\log n} \frac{du}{u} = \log\log n + O(1),$$

and the second one becomes

$$\int_{\log 3/2}^{\log n} \frac{du}{u^2} = \frac{1}{\log 3/2} - \frac{1}{\log n} = O(1).$$

So

$$\sum_{p \le n} \frac{1}{p} = \log\log n + O(1).$$

Therefore, the number of array accesses done by the sieve is

$$n \log\log n + O(n).$$

We can give this result the following interpretation. The average work per prime is about

$$\frac{n \log\log n}{n/\log n} = \log n \log\log n.$$

This is an example of an *amortized* complexity bound.

Partial summation.

This is a discrete analog of integration by parts. Assume for the moment that f and g are continuous functions. Then

$$\begin{aligned}\sum_{k=1}^{n} f(k)[g(k)-g(k-1)] &= \int_0^n f(x)dg(\lfloor x\rfloor) \\ &= fg|_0^n - \int_0^n g(\lfloor x\rfloor)df(x)\end{aligned}$$

Since f is continuous we have

$$\begin{aligned}\int_0^n g(\lfloor x\rfloor)df(x) &= \sum_{k=0}^{n-1}\int_k^{k+1} g(\lfloor x\rfloor)df(x) \\ &= \sum_{k=0}^{n-1} g(k)\int_k^{k+1} df(x) \\ &= \sum_{k=0}^{n-1} g(k)[f(k+1)-f(k)]\end{aligned}$$

We can write the resulting formula compactly as

$$\sum_{k=1}^{n} f(k)\bigtriangledown g(k) = [fg]_0^n - \sum_{k=0}^{n-1} g(k)\bigtriangleup f(k).$$

Mnemonic: The forward difference operator $\bigtriangleup$ goes up to get its arguments, and the backward difference operator $\bigtriangledown$ goes down.

If you find this confusing, you are not alone. One of the strengths of Stieltjes integrals is that the integration by parts formula looks just like the familiar formula from calculus.

The result actually holds for any two functions f and g, since you can always interpolate continuous functions between the values f and g take at integers.

Euler-Maclaurin summation

If a and b are integers, we should expect, heuristically at least, that

$$\sum_{a<k\le b} f(k) \approx \int_a^b f(x)dx.$$

Why? The sum on the left is a Riemann sum for the integral, so if f doesn't vary too much it will be a good approximation to it.

The Euler-Maclaurin formula gives a way to estimate the error in this approximation. In a nutshell, the idea is to write the sum as a Stieltjes integral and repeatedly integrate by parts.

Assume that f is C^∞ (all derivatives exist) and $a < b$ (integers). Then

$$\begin{aligned}
\sum_{a<k\le b} f(k) &= \int_a^b f(x)d\lfloor x\rfloor \\
&= \int_a^b f(x)dx + \int_a^b f(x)d(\lfloor x\rfloor - x + 1/2) \\
&= \int_a^b f(x)dx + [f(x)(\lfloor x\rfloor - x + 1/2)]_a^b - \int_a^b f'(x)(\lfloor x\rfloor - x + 1/2)dx \\
&= \int_a^b f(x)dx + \frac{f(b)-f(a)}{2} + \int_a^b f'(x)B_1(\{x\})dx,
\end{aligned}$$

where we have written

$$B_1(t) = t - 1/2, \qquad \{x\} = x - \lfloor x\rfloor.$$

B_1 is a Bernoulli polynomial, and the subscript 1 is a warning that more are on the way.

The graph of $B_1(\{x\})$ has a sawtooth shape:

In particular, $|B_1(\{x\})| \le 1/2$.

Let's see what this gives us for the sum of the first n d-th powers.

$$\sum_{k=1}^{n} k^d = \sum_{0<k\le n} k^d = \int_0^n x^d dx + \frac{n^d}{2} + d\int_a^b x^{d-1}B_1(\{x\})dx,$$

which equals

$$\frac{n^{d+1}}{d+1} + O(n^k).$$

Since $B_1(\{x\})$ averages to 0, we guess that the remainder is small compared to the second term $n^d/2$. Further analysis will verify this.

We now want to get a better estimate for the error

$$\int_a^b f'(x)B_1(\{x\})dx$$

using integration by parts.

We need to express B_1 as the derivative of something. We have

$$\int B_1(t)dt = \int (t - 1/2)dt = 1/2(t^2 - t + C),$$

and we choose C to make this new function integrate to 0 over $[0, 1]$. This implies that we should have $C = 1/6$, and we set

$$B_2(t) = t^2 - t + 1/6.$$

With this definition in hand, we have

$$\int_a^b f'(x)B_1(\{x\})dx$$

$$= \frac{1}{2}\int_a^b f'(x)dB_2(\{x\}) = \frac{1}{2}\left[B_2(\{x\})f'(x)\right]_a^b - \frac{1}{2}\int_a^b f''(x)B_2(\{x\})dx$$

The graph of $B_2(\{x\})$ is wavy, with cusps at integer points:

Observing that $B_2(\{x\}) = 1/6$ when x is an integer, we get a second formula:

$$\sum_{a<k\leq b} f(k) = \int_a^b f(x)dx + \frac{f(b) - f(a)}{2} + \frac{f'(b) - f'(a)}{12} - \int_a^b \frac{B_2(\{x\})f''(x)}{2}dx.$$

This process can be repeated, using further polynomials $B_3, B_4, \ldots$, which we now define.

Bernoulli polynomials and Bernoulli numbers

These are defined by recursion.

Set $B_0(t) = 1$.

$B_m(t) = m\int B_{m-1}(t)dt + C$, where the constant C is chosen to make $\int_0^1 B_m(t)dt$ vanish.

The m-th *Bernoulli number* is the constant term of $B_m(t)$. That is,

$$B_m = B_m(0).$$

Here are the first few Bernoulli polynomials.

m	$B_m(t)$	B_m
1	$t - 1/2$	$-1/2$
2	$t^2 - t + 1/6$	$1/6$
3	$t^3 - (3/2)t^2 + t/2$	0
4	$t^4 - 2t^3 + t^2 - 1/30$	$-1/30$
5	$t^5 - (5/2)t^4 + (5/3)t^3 - t/6$	0

Abramowitz and Stegun (p. 809) tabulate these for $m \leq 15$.

It can be shown that B_m vanishes for odd m, except for B_1.

The following estimate is useful:

$$\left| \frac{B_m(\{x\})}{m!} \right| \leq \frac{4}{(2\pi)^m}.$$

Euler-Maclaurin Summation

This is actually a family of formulas, each containing a remainder given by an integral.

After m applications of integration by parts, we get

$$\sum_{a<k\leq b} f(k) = \int_a^b f(x)dx + \sum_{j=1}^{m} (-)^j \frac{B_j}{j!} \left[f^{(j-1)}(x) \right]_a^b + R_m,$$

where

$$R_m = (-)^{m+1} \frac{1}{m!} \int_a^b B_m(\{x\}) f^{(m)}(x)dx.$$

Summation of the first n d-th powers.

We want to evaluate

$$\sum_{k=1}^{n} k^d.$$

Let $a = 0$, $b = n$, and $f(t) = t^k$. Choosing $m = k + 1$ to make R_m vanish, we get the series

$$\sum_{k=1}^{n} k^d = \frac{n^{d+1}}{d+1} + \sum_{j=1}^{d} (-)^j \binom{d}{j-1} \frac{B_j}{j} n^{d-j+1}.$$

The index only goes to d because the d-th derivative is constant, making the difference vanish.

Example:

$$\sum_{k=1}^{n} k^3 = \frac{n^4 + 2n^3 + n^2}{4} = \left(\sum_{i=1}^{n} i \right)^2 .$$

(The last identity is due to Nicomachus, circa 100 A.D.)

Note that the numerator of this "summation polynomial" is identical to the Bernoulli polynomial, except for the plus sign on the term of next-to-highest degree. This always happens.

Bernoulli was extremely proud of this result, and remarked that he used it to sum the first thousand tenth powers, in seven and one half minutes. (*Intra semi-quadrantem horae.*)

Stirling's Formula

Using Euler's summation formula (Euler-Maclaurin for $m = 1$), we can easily prove a computer scientist's version of Stirling's formula.

Start with $\log n! = \sum_{1<k\leq n} \log k$. This equals

$$\int_1^n \log t \; dt + \left[\frac{\log n}{2} \right]_1^n + R,$$

where

$$R = \int_1^n \frac{1}{t} B_1(\{t\}) dt.$$

Evaluating the integral, we get

$$\log n! = n \log n - n + 1 + \frac{\log n}{2} + R.$$

We now show that $R = \Theta(1)$. Observe that

$$R = \sum_{r=1}^{n-1} \int_r^{r+1} B_1(\{t\}) \frac{dt}{t}$$

Split each integral in the sum into an integral from r to $r + 1/2$ (call this I_1) and an integral from $r + 1/2$ to r (call this I_2). On $(r, r+1)$, the graph of $B_1(\{t\})$ makes two congruent triangles; call their common area A. Then

$$\frac{1}{r+1} A \leq I_2 \leq \frac{1}{r} A,$$
$$-\frac{1}{r} A \leq I_1 \leq -\frac{1}{r+1} A.$$

This gives (since $A = 1/4$)

$$\left| \sum_{r=1}^{n-1} \int_r^{r+1} B_1(\{t\}) \frac{dt}{t} \right| \le \frac{1}{4r(r+1)},$$

so

$$|R| \le \frac{1}{4} \sum_{r=1}^{\infty} \frac{1}{r(r+1)} = \frac{1}{4}.$$

Exponentiating, we conclude that

$$n! = \Theta\left(\frac{n^{n+1/2}}{e^n} \right).$$

From the bound on R, the implied constant is between 2 and 3.5. It is actually $\sqrt{2\pi} = 2.505528\ldots$.

Notes

The Bernoulli quote is in Chrystal, Textbook of Algebra, v. 2, p. 233.

Extending the Riemann zeta function to the left of $s = 1$ is another nice example. See H. M. Edwards, Riemann's Zeta Function, Section 6.4.

For the history of the Euler-Maclaurin formula, see Chapter IV of N. Bourbaki, Fonctions d'une Variable Réelle, pp. 155-159.

Lecture 19

Topic du jour: Asymptotic integration and summation

References: J. Dieudonne, *Infinitesimal Calculus*, Chap. III. See also G. H. Hardy, *Orders of Infinity*.

What's this about?

Our goal is to quickly find asymptotic forms for integrals such as

$$\int_a^x t^2 \log\log t dt,$$

and similar sums.

In particular, we'd like to justify the following heuristic idea: relative to powers of t, $\log\log t$ grows very slowly, so the following approximation should be valid:

$$\int_a^x t^2 \log\log t \; dt \sim \log\log x \int_a^x t^2 \; dt \sim \frac{x^3 \log\log x}{3}.$$

Formal rules for making such approximations were worked out by du Bois-Reymond in the late 1800's, and later put on a rigorous foundation by Hardy and others.

A striking feature of this theory is the classification of functions by their logarithmic derivatives.

Some easy estimates

Let f, g be continuous functions, and assume that g is ultimately positive, with $\int_a^\infty g = \infty$. Then:

1. If $f = O(g)$, then $\int_a^x f = O(\int_a^x g)$.
2. If $f = o(g)$, then $\int_a^x f = o(\int_a^x g)$.
3. If $f \sim g$, then $\int_a^x f \sim \int_a^x g$.

Note that there are no corresponding rules for differentiation.

Example: Let $f(x) = \sin(x^2)$. Then $f(x) = O(1)$, but $f'(x) = 1 + 2x\cos(x^2)$ is unbounded.

Asymptotic integrals for functions that look like powers

It's common to have integrals of the form $\int \gamma(x) x^\mu dx$, where γ changes slowly.

Examples: $\int x \log x$, $\int x/(\log x)$, etc.

Theorem: Let f be a continuously differentiable positive function, for which

$$\frac{d}{dx} \log f(x) (= f'/f) \sim \frac{\mu}{x}.$$

(We call μ the *order* of f.) Then if $\mu > -1$, $\mu \neq 0$, we have

$$\int_a^x f(t)dt \sim \frac{xf(x)}{\mu+1}.$$

The hypothesis is that f is approximately x^μ. Indeed, since $(\log g)' \sim \mu/x$, 3.ãbove implies that $\log g \sim \mu \log x$, making $g = x^{\mu+o(1)}$.

Proof: Use integration by parts. We have

$$\begin{aligned}\int_a^x f(t)dt = &\ xf(x) - af(a) - \int_a^x tf'(t)dt \\ \sim &\ xf(x) - \mu \int_a^x f(t)dt\end{aligned}$$

(using $tf' \sim \mu f$ and 3. above), so

$$(1+\mu)\int_a^x f(t)dt \sim xf(x).$$

Example (from the beginning of the lecture). Let $f(x) = x^2 \log\log x$. Then

$$\frac{f'}{f} = (\log f)' \sim \frac{2}{x},$$

so this function has order 2. The theorem tells us that for sufficiently large a,

$$\int_a^x t^2 \log\log t \; dt \sim \frac{x^3 \log\log x}{3}.$$

The theorem can be extended to handle functions of order 0 ($\mu = 0$). This needs the stronger hypothesis that $g'/g = o(1/x)$.

The proof is the same, except that $\int_a^x tf' = \int_a^x to(f/t) = o\left(\int_a^x f\right)$ is now asymptotically negligible.

Example: the logarithm integral (beloved of analytic number theorists).

$$\int_2^x \frac{dt}{\log t} \sim \frac{x}{\log x},$$

since the logarithmic derivative of $1/(\log x)$ is $-1/(x \log x) = o(1/x)$.

When $\mu = -1$, the best thing to do is to substitute $u = \log t$ and continue. Example:

$$\int_a^x \frac{\log\log t}{t} dt = \int_{\log a}^{\log x} \log u du \sim \log x \log\log x.$$

Handling functions that are large compared to powers.

Theorem: If $f, f' > 0$ and f is C^2, then

$$\int_a^x f(t)dt \sim \frac{f(x)}{f'/f(x)} = \frac{f(x)^2}{f'(x)}$$

whenever $f'/f \to \infty$.

This result says that if f is large compared to any power of x, we get the asymptotic integral by dividing by the logarithmic derivative. Note that the hypothesis is equivalent to $f/f' = o(1)$.

For a proof, see Dieudonné, p. 82.

Example: Let $f(x) = e^{x^2}$. (This is large, if any function is!) Then $(\log g)' = (x^2)' = 2x$, so

$$\int_a^x e^{t^2} dt \sim \frac{e^{x^2}}{2x}.$$

Tails of convergent integrals.

Suppose that $\int_a^\infty f(t) < \infty$. Then, with suitable hypotheses, we have

$$\text{order}(f) < -1 \qquad \Longrightarrow \qquad \int_x^\infty f(t) \sim -\frac{xf(x)}{\mu + 1}$$

and

$$\text{order}(f) = -\infty \qquad \Longrightarrow \qquad \int_x^\infty f(t) \sim -\frac{f(x)^2}{f'(x)}$$

For example, for the tail of the normal probability distribution we have

$$\frac{1}{\sqrt{2\pi}} \int_x^\infty e^{-t^2/2} dt \sim \frac{1}{\sqrt{2\pi}} \frac{e^{-x^2/2}}{x}$$

Deriving asymptotic series

For integrals of the type discussed here, we can usually obtain asymptotic series by repeatedly expressing the error as an integral.

Let's use our example of $f(x) = x^2 \log\log x$ again. Write

$$\int_a^x t^2 \log\log t dt = \frac{x^3 \log\log x}{3} + e(x),$$

where e is the error. Then

$$\frac{d}{dx}\left(\int_a^x t^2 \log\log t dt - \frac{x^3 \log\log x}{3} \right) = e'(x),$$

so

$$e'(x) = -\frac{x^2}{3\log x}.$$

Therefore,

$$e(x) \sim -\frac{x^3}{9\log x},$$

and we have the more accurate result

$$\int_a^x t^2 \log\log t dt \sim x^3 \left(\frac{\log\log x}{3} - \frac{1}{9\log x}\right).$$

If we repeat this process, we get an asymptotic series

$$\int_a^x t^2 \log\log t dt \sim \frac{x^3}{3}\left(\log\log x - \sum_{n\geq 1}\frac{(n-1)!}{(3\log x)^n}\right)$$

Note: Since $x^3/(3\log x) = x^3/\log(x^3)$, we have also found an asymptotic series for the logarithm integral:

$$\int_2^x \frac{dt}{\log t} \sim x\left[\frac{1}{\log x} + \frac{1}{(\log x)^2} + \cdots + \frac{(k-1)!}{(\log x)^k} + \cdots\right]$$

Sums

Unfortunately, the naive idea of replacing a sum by an integral doesn't always work.

Example: Let $S(n) = \sum_{i=1}^n i2^i$.

We can sum this easily with a trick.

$$\begin{aligned} S &= \sum_{i=1}^n i2^i \\ 2S &= \sum_{i=1}^n i2^{i+1} = \sum_{i=2}^{n+1}(i-1)2^i \end{aligned}$$

Subtract and get

$$-S = S - 2S = \sum_{i=1}^n 2^i - n2^{n+1},$$

so $S = n2^{n+1} - 2(2^n - 1) \sim n2^{n+1}$.

On the other hand,

$$\int_1^n x2^x dx = \int_1^n xe^{x\log 2}\, dx \sim \frac{n2^n}{\log 2}.$$

So we got the right growth rate but the wrong leading constant. The problem is that you can't get rid of $x2^x$ by differentiation, so the "error term" in Euler's summation formula will be large.

A correction was worked out by Hardy.

Let a be a smooth positive function, for which $\sum_{n\geq 1} a_n$ diverges.

If $a'/a \to \infty$, then

$$\sum_{i=1}^{n} a_i \sim a_n$$

(the last term dominates).

If $a'/a \sim \mu$ – this says that $a(t)$ equals $e^{\mu t}$ times a smaller factor – then

$$\sum_{i=1}^{n} a_i \sim \frac{\mu}{1-e^{-\mu}} \int_1^n a(t)dt.$$

If $a'/a \to 0$, then

$$\sum_{i=1}^{n} a_i \sim \int_1^n a(t)dt.$$

[Exact assumptions should be stated here.]

Example: Average-case analysis of the number of comparisons done by quicksort leads to the sum

$$\sum_{i=1}^{n} \frac{i-1}{i(i+1)}.$$

For $a(x) = (x-1)/(x(x+1))$, we have $a'/a \sim -1/x$ (third case above), so

$$\sum_{i=1}^{n} \frac{i-1}{i(i+1)} \sim \int_1^n \frac{x-1}{x(x+1)}dx = \log(n+1) - 2\log 2 + \log(1+1/n) \sim \log n.$$

Notes.

There is an abstract approach to all this, using the concept of a Hardy field. See N. Bourbaki, Fonctions d'une Variable Réele, Chap. 5, pp. 107-126.

The German (!) mathematician Paul du Bois-Reymond is probably best known for having proved that the Fourier series of a continuous function can diverge.

The ad hoc assumptions for functions should be replaced by something uniform.

Lecture 20

Topic du jour: A crash course on complex analysis

Reference: For a bird's eye view of complex variables see Appendix V of R. C. Buck, Advanced Calculus, or Appendix A.2 of Purdom and Brown. A good "applied math" book is R. V. Churchill, Complex Variables with Applications. Purists may prefer L. V. Ahlfors, Complex Analysis.

What is complex analysis and why do we need it?

Our goal is to study the growth rate of sequences by investigating properties of their generating functions. These generating functions are most fruitfully viewed as functions of a complex variable.

We'll study a class of functions, called *analytic*, that allow us to generalize the ideas of calculus to complex variables. Analytic functions are extremely well-behaved and easy to work with. They include rational functions and complex extensions of the familiar elementary functions: exponential, logarithm, square root, etc.

Two themes of our work will be: a) the connection between the growth rate of a sequence and properties of its generating function, and b) representation of sequence elements using integrals.

Complex numbers

If x and y are real numbers, then $z = x + iy$ is a complex number. These are manipulated using the rules of ordinary algebra, together with the rule $i^2 = -1$.

The *absolute value* of z is

$$|z| = \sqrt{x^2 + y^2}.$$

(This is just the usual Euclidean length.)

We will also need the polar representation

$$z = re^{i\theta},$$

where $r = |z|$ and θ is real.

A sequence $z_1, z_2, \ldots$ converges to z if the real seqence $|z_n - z| \to 0$. An infinite sum converges when the sequence of partial sums converges, just as with real variables.

Analytic functions

Let D be an open set in the complex plane, with $z \in D$. A function $f : D \to \mathbf{C}$ is called *differentiable at* z if

$$\lim_{h \to 0} \frac{f(z+h) - f(z)}{h}$$

exists, giving the same value no matter how h approaches zero.

Example: Let $f(z) = z^n$. Then

$$(z+h)^n - z^n = nz^{n-1}h + h^2 g(z),$$

where g is a polynomial. Therefore, the derivative exists and equals nz^{n-1}.

Non-example: Let $f(z) = |z|$. In the radial direction, the derivative is 1, whereas in the tangential direction, the derivative is 0.

Definition. The function f is *analytic on* D if $f'(z)$ exists for every $z \in D$.

Theorem: Let $f(z) = u(x,y) + iv(x,y)$ be analytic. Then the *Cauchy-Riemann* equations

$$\frac{\partial u}{\partial x} = \frac{\partial v}{\partial y}, \qquad \frac{\partial v}{\partial x} = -\frac{\partial u}{\partial y}$$

hold.

To prove this, compare the limits of $[f(z+h) - f(z)]/h$ as $h \to 0$ horizontally and vertically.

If you know about the Fréchet derivative, you can give this result the following interpretation. Considered as a function from $\mathbf{R}^2$ to itself, the derivative of f must take the form $\begin{pmatrix} a & b \\ -b & a \end{pmatrix}$. This shows that analyticity is stronger than just having a derivative in the real sense.

Analytic functions are extremely well behaved. We list some properties without proof, which are true when f is analytic on the open set D.

Considered as a function from D to $\mathbf{R}^2$, f is C^∞. (This means that any conceivable partial derivative exists and is continuous.)

All higher derivatives of f exist, in the complex sense.

If f vanishes infinitely often on some bounded subset of D, then $f = 0$ *everywhere* on D. (Intuitively, the function can be "cloned" from any local piece, no matter how small.)

$|f|$ cannot have a maximum at any interior point of D.

Singularities

Let D be a disk around a. If f is analytic in $D - \{a\}$ but cannot be extended to an analytic function on D, then we call the point a a *singularity*.

The simplest kind of singularity involves division by 0. If

$$f(z) = \frac{g(z)}{(z-a)^n},$$

where g is analytic near a, we say that f has a pole of order n at a.

Power series.

Suppose f is analytic for $|z| < R$. Then at all points inside this disk, we have

$$f(z) = f(0) + f'(0) + f''(0)z^2/2 + \cdots,$$

that is, f is the sum of its Taylor series. Inside the disk this series converges absolutely and you can differentiate and integrate it term by term.

Conversely, suppose that

$$f(z) = a_0 + a_1 z + \cdots$$

is a power series. If $|a_i| = O(r^i)$ for some positive real number r, then f represents an analytic function for $|z| < 1/r$.

More precisely, we define a number R by

$$1/R = \limsup \sqrt[n]{|a_n|}.$$

This is called the *radius of convergence* of the series. The series gives an analytic function for $|z| < R$ and diverges for $|z| > R$. Convergence on the circle is neither assured nor forbidden.

Generally, a power series will converge until it is prevented from doing so by some singularity.

Example:

$$f(z) = 1/(1+z^2) = 1 - z^2 + z^4 + \cdots$$

converges for $|z| < 1$. Note that there are poles on the unit circle, at $\pm i$.

Recalling that the radius of convergence of a power series is determined by the growth rate of its coefficients, we see that this growth rate is tied to the singularities of the function represented by the series. Even if the coefficients are all real, the singularities may not be, as the above example shows.

Integrals

This works just like integration in the real case, except that we must pick a path over which to perform the integration.

The easiest case to consider is the integral of a complex function of a real variable, say

$$\int_a^b z(t)dt.$$

In this case we can imagine walking from a to b, carrying a TV screen that records the value of $z(t)$ for each point t. The integral will be the average of the values on the screen, times the length of the path. To make this rigorous, we let $z = x + iy$ with x, y real, and define

$$\int_a^b z(t)dt = \int_a^b x(t)dt + i\int_a^b y(t)dt.$$

All other cases can be reduced to this one. Suppose $\gamma : [0,1] \to \mathbf{C}$ be a smooth curve. Then we define

$$\int_\gamma f(z)dz = \int_0^1 f(\gamma(t))\gamma'(t)dt.$$

This reduces the integration of a complex function to the evaluation of two real integrals.

As with real line integrals it can be shown that all smooth parameterizations of the curve give the same result.

Example: Let us integrate z^m over the circle given by $re^{2\pi it}$, where $0 \le t \le 1$. We get

$$\int_\gamma z^m dz = 2\pi i r^{m+1} \int_0^1 e^{2\pi i(m+1)t} dt = \begin{cases} 0, & \text{if } m \neq -1; \\ 2\pi i, & \text{if } m = -1. \end{cases}$$

For $m \ge 0$ this is a special case of the residue theorem which will be discussed in the next lecture.

Lecture 21

Topic du jour: Residue theorem, residue integrals.

References: See last lecture.

Recall our definitions:

The function f is analytic (in an open set D) if $f'(z)$ exists for each $z \in D$.

Complex integrals are line integrals. If $\gamma : [0,1] \to \mathbf{C}$ is a path, then

$$\int_\gamma f(z)dz = \int_0^1 f(\gamma(t))\gamma'(t)dt.$$

Dependence on path

Any time you deal with line integrals the question of path dependence arises. To see that an integral can depend on the path, consider

$$\int_1^{-1} \frac{dz}{z},$$

over the paths

$$\gamma_+ = \text{ semicircle of radius 1 over the orgin}$$

and

$$\gamma_- = \text{ semicircle of radius 1 under the orgin .}$$

We will let $\eta = \gamma_+ - \gamma_-$, so that η winds once around the origin in the counterclockwise direction.

We proved last time that

$$\int_{\gamma_+ - \gamma_-} \frac{dz}{z} = \int_\eta \frac{dz}{z} = 2\pi i,$$

so

$$\int_{\gamma_+} \frac{dz}{z} = \int_{\gamma_-} \frac{dz}{z} + 2\pi i.$$

We would like a condition that prevents such annoyances.

Cauchy's Theorem

Theorem: Let γ be a curve forming the boundary of the open set D, and assume that f is analytic on some open set containing $\bar{D}$. Then $\int_\gamma f(z)dz = 0$.

This follows from the Stokes theorem:

$$\int_{\partial D} f dz = \int_D d(f dz).$$

To see this, use the Cauchy-Riemann equations. From

$$f = u + iv, \qquad dz = dx + idy,$$

we get

$$f dz = (u dx - v dy) + i(v dx + u dy).$$

Then

$$d(u dx - v dy) = u_y dx dy - v_x dx dy = (u_y + v_x) dx dy = 0.$$

Similarly,

$$d(v dx + u dy) = 0.$$

So $d(f dz) = 0$, making the integral 0.

One way to think of this is to say that the integral of an analytic function along two different paths with the same endpoints must give the same result, provided that the two paths bound a region in which the function is analytic.

The Residue Theorem

Theorem: Let f be analytic in the open set D. Assume that the closed disk of radius r around a lies inside D. Then we have

$$\frac{1}{2\pi i} \int_{|z|=r} \frac{f(z)dz}{z-a} = f(a).$$

Idea of proof: We can deform the path into a small circle around a without changing the value of the integral. Then

$$\int_{|z|=\epsilon} \frac{f(z)dz}{z-a} \approx f(a) \int_{|z|=\epsilon} \frac{dz}{z-a} = 2\pi i f(a).$$

More generally, we have

$$\frac{1}{2\pi i} \int_{|z|=r} \frac{f(z)dz}{(z-a)^{n+1}} = \frac{1}{n!} f^{(n)}(a).$$

This gives a way of extracting the n-th term of a power series, by doing an integral.

Computing integrals via residues

This isn't really part of our course, but it is such a powerful tool that we mention it here.

Consider

$$I = \int_0^{2\pi} \frac{d\theta}{a + \cos\theta},$$

with $a > 1$.

The basic idea is to express the integral as a complex integral over a closed path, and evaluate it by looking at the singularities inside the path.

We will use the unit circle, parametrized by $z = e^{i\theta}$. On the circle we have $dz = iz d\theta$, and

$$\cos\theta = \frac{e^{i\theta} + e^{-i\theta}}{2} = \frac{z + z^{-1}}{2}.$$

So

$$I = -4\pi \times \frac{1}{2\pi i} \int_{|z|=1} \frac{dz}{z^2 + 2az + 1}.$$

Next, we examine the singularities of the integrand. We have

$$z^2 + az + 1 = (z - \alpha)(z - \beta)$$

with $\alpha\beta = 1$. The numbers α and β are real numbers whose product is 1. Since $a > 1$ we can assume that

$$\alpha < 1 < \beta$$

By the residue theorem and the quadratic formula

$$\begin{aligned} I &= -4\pi \times \frac{1}{2\pi i} \int_{|z|=1} \frac{dz}{(z - \alpha)(z - \beta)} \\ &= \frac{-4\pi}{\alpha - \beta} \\ &= \frac{2\pi}{\sqrt{a^2 - 1}}. \end{aligned}$$

Notes

Homology vs. homotopy. Some people state the hypothesis for Cauchy's theorem as a requirement that the path be shrinkable to a point (null homotopic), whereas we required that it be the boundary of a region (homologous to zero). In the plane these are the same, but on more complicated surfaces the second is more general. For example, a pretzel with two holes has a bounding cycle that cannot be shrunk to a point.

Some better (i.e. more to the point of this course) applications should be included.

"Rice's method" for alternating binomial sums is a good application of residue calculus. See H. Prodinger, How to Advance on a Stairway by Coin Flippings, in G. E. Bergum et al (eds), Applications of Fibonacci Numbers, vol. 5, pp. 473-479, 1993, and references therein.

G. G. Comisar, On the evaluation of finite sums by residues, AMM 67, 1960, pp. 775-776.

Lecture 22

Topic du jour: Darboux method for estimating sequence growth rates

References: GK 4.3.1. I don't know a source for the Bernoulli number asymptotics. For 3-way sorting, see Knuth, v. 3, Ex. 3 of 5.3.1.

The main idea

As a general rule, a power series will converge until it is prevented from doing so by a singularity.

We can think of power series coefficients as the the "cause" of the singularity, in some sense. (Why not? They determine everything you want to know about the function.) Different growth rates cause different types of singularities.

The idea behind Darboux's method is to find a "comparison function" to match the nearest singularity. Then its power series coefficients will be a good approximation to coefficients of the original power series.

We will use the following result. If $f(z) = \sum a_n z^n$ is analytic for $|z| \le r$, then $|a_n| \le r^{-n}$ for all sufficiently large n. (Exercise. It is easy to prove using Hadamard's radius of convergence formula).

Example 1: Bernoulli numbers

These come up in the Euler-Maclaurin formula so it would be nice to know how big they are.

It can be shown that

$$f(z) = \frac{z}{e^z - 1} = 1 - \frac{1}{2}z + \frac{1}{12}z^2 - \frac{1}{720}z^4 + \cdots + \frac{B_n z^n}{n!} + \cdots$$

where B_n is the n-th Bernoulli number.

This is an example of an exponential generating function. The idea is that B_n is too large for the ordinary generating function to be useful, so we "mollify" it by dividing by $n!$.

$f(z)$ blows up when $z = \pm 2k\pi i$, $k \neq 0$. Note that the apparent singularity at $z = 0$ is removable, since $e^z - 1 = z(1 + z/2 + z^2/6 + \cdots)$.

Crude estimates for Bernoulli numbers

The sequence $\{B_n\}$ is not bounded, for if it were, f would be defined everywhere (by comparison with the exponential series).

The function f is analytic for $|z| \le r < 2\pi$, so we can assert that for any $\epsilon > 0$,

$$|B_n| = O\left(\frac{n!}{(2\pi - \epsilon)^n}\right).$$

How does $f(z)$ behave near $2\pi i$?

Let $z = w + 2\pi i$. Then

$$f = \frac{w + 2\pi i}{e^w - 1} \sim \frac{2\pi i}{e^w - 1} \sim \frac{2\pi i}{w} = \frac{2\pi i}{z - 2\pi i}.$$

Similarly, near $-2\pi i$ we have

$$f \sim \frac{-2\pi i}{z + 2\pi i}.$$

Asymptotics for Bernoulli numbers

This suggests we can, for the purpose of approximating its power series coefficients, replace f by the rational function

$$g(z) := \frac{2\pi i}{z - 2\pi i} - \frac{2\pi i}{z + 2\pi i} = -\left(\frac{1}{1 - z/2\pi i} + \frac{1}{1 + z/2\pi i}\right)$$

We note

$$g(z) = \sum_{n \geq 0} (-)^{n+1} \frac{2z^{2n}}{(2\pi)^{2n}},$$

and that $f(z) - g(z)$ has the radius of convergence 4π.

Therefore,

$$\frac{B_{2n}}{(2n)!} = (-)^{n+1} \frac{2}{(2\pi)^{2n}} + O\left(\frac{1}{(4\pi - \epsilon)^{2n}}\right),$$

which implies

$$B_{2n} \sim (-)^{n+1} \frac{2(2n)!}{(2\pi)^{2n}}.$$

A series for Bernoulli numbers

We can continue this process and get

$$\frac{B_{2n}}{(2n)!} \sim (-)^{n+1} \frac{2}{(2\pi)^{2n}} \left(1 + 2^{-2n} + 3^{-2n} + \cdots + m^{-2n}\right),$$

in the sense that the relative error is $o(m^{-2n})$.

This is for fixed m, $n \to \infty$. What about the other way? As $m \to \infty$, the series in parentheses converges, to the value

$$\zeta(2n) := \sum_{m \geq 1} \frac{1}{m^{2n}}.$$

It can be shown that there is an identity

$$\frac{B_{2n}}{(2n)!} = (-)^{n+1} \frac{2}{(2\pi)^{2n}} \zeta(2n).$$

(Euler?)

Using this series we can evaluate the Riemann zeta function at even positive integers. We get

$$\begin{aligned}\zeta(2) &= \frac{B_2}{2!}\frac{(2\pi)^2}{2} = \frac{\pi^2}{6},\\ \zeta(4) &= -\frac{B_4}{4!}\frac{(2\pi)^4}{2} = \frac{\pi^4}{90},\\ \zeta(6) &= \frac{B_6}{6!}\frac{(2\pi)^6}{2} = \frac{\pi^6}{945},\end{aligned}$$

and so on.

Example 2: Sorting with ties allowed.

The usual theory of sorting deals with inputs that are distinct. Suppose we allow ties? In many different ways can we order n keys, with duplications allowed?

Example: if we have two keys a and b, the different orderings are

$$a < b, \qquad a = b, \qquad b < a.$$

Let T_n be the number of different orderings. We can express this using Stirling numbers. We have

$$T_n = \sum_{k=1}^{n} \left\{ {n \atop k} \right\} k!,$$

since we must choose k equivalence classes, then order them.

Using the recurrence relation for Stirling numbers, it's possible to compute T_n using $O(n^2)$ arithmetic operations. Starting with $T_0 = 1$, the sequence goes

$$1, 1, 3, 13, 75, 541, 4683, 47293, 545835, \ldots\ .$$

Unfortunately there doesn't seem to be a closed form for this, so we need a different attack.

Finding a generating function

We have the recurrence relation

$$T_n = \sum_{k=1}^{n} \binom{n}{k} T_{n-k}.$$

(Proof: We can choose an equivalence class for the smallest keys of size k in $\binom{n}{k}$ ways. For each of these we have T_{n-k} ways to order the remaining elements.)

We rewrite this as

$$\frac{T_n}{n!} = \sum_{k=1}^{n} \frac{1}{k!}\frac{T_{n-k}}{(n-k)!},$$

and the convolution suggests that we should use the exponential generating function

$$F(z) = \sum_{n\geq 0} \frac{T_n z^n}{n!}.$$

Indeed, we have

$$F(z) - 1 = \left(\sum_{k\geq 1} \frac{z^k}{k!}\right)\left(\sum_{m\geq 0} \frac{T_m z^m}{m!}\right) = (e^z - 1)F(z).$$

This is readily solved to get

$$F(z) = \frac{1}{2 - e^z}.$$

Where are its singularities?

$F(z)$ is analytic except when denominator vanishes, so its singularities are at

$$z = \log 2 + 2k\pi i, \qquad k \in \mathbf{Z}.$$

Asymptotics for T_n.

Put $z = \log 2 + w$, then near $w = 0$ we have

$$\begin{aligned} F(z) &= -\frac{1}{2(e^w - 1)} \\ &= -\frac{1}{2w(1 + O(w))} \\ &= -\frac{1}{2w} + O(1). \end{aligned}$$

Then

$$G(z) := -\frac{1}{2w} = \frac{1}{2(\log 2 - z)} = \frac{1}{2\log 2}\left(1 + \cdots + \frac{z^n}{(\log 2)^n} + \cdots\right)$$

is a good approximation to $F(z)$ near $z = \log 2$, in the sense that $F - G$ is analytic for

$$|z| < R = |\log 2 + 2\pi i| = 6.321303...$$

This gives us

$$\frac{T_n}{n!} = \frac{1}{2(\log 2)^{n+1}} + O(6^{-n}),$$

an excellent approximation. For $n = 3$ this yields 12.99629..., which compares nicely to the true value of 13.

Continuing this process, we can get more accurate approximations. You can verify that the level ℓ approximation is given by

$$\frac{T_n}{n!} = \frac{1}{2(\log 2)^{n+1}} + \sum_{k=1}^{\ell} \mathrm{Re} \frac{1}{(\log 2 + 2k\pi i)^{n+1}} + O\left((6\ell+6)^{-n}\right).$$

Here is some numerical data.

n	exact	$\ell = 0$	1	2	3
1	1	1.040684490	1.016260561	1.009985496	1.007182410
2	3	3.002780708	3.000217810	3.000052714	3.000019918
3	13	12.99629050	12.99969112	12.99992448	12.99997137
4	75	74.99873542	74.99997698	74.99999767	74.99999952
5	541	541.0015182	541.0000315	541.0000030	541.0000003

Notes

Darboux's original paper is in J. Math. Pures Appl., ser. 3, v. 4 (1878) pp. 5-56. He says (p. 16) that the use of poles (as in this lecture) is a "corollary of the beautiful results of Cauchy," and then goes on to discuss algebraic singularities.

For explicit bounds on Bernoulli numbers, see C. D'Aniello, Rend. Circ. Mat. Palermo (2) 43, 1994, no. 3, pp. 329-332. (MR 96f:11030)

Wilf (Generatingfunctionology) calls T_n the number of ordered partitions of $\{1, \ldots, n\}$.

Another good example for Darboux is the theory of success runs in coin flips. See Feller I, p. 341.

Darboux works very well for the parking problem. After you subtract out a "principal part" corresponding to a double pole, you get an entire function.

Lecture 23

Topic du jour: Solving regular and context-free counting problems

References: Flajolet/Sedgewick Chap. 7. Hopcroft and Ullman (for background on language theory).

Rational and algebraic functions

Rational function: ratio of polynomials, e.g.

$$f(z) = \frac{p(z)}{q(z)}$$

Analytic except when $q(z) = 0$.

Singularities are always poles.

Algebraic function: solution $y(z)$ to a polynomial equation

$$P(y, z) = a_n(z)y^n + \ldots + a_1(z)y + a_0(z) = 0$$

in which the a_i are rational functions. After clearing denominators, we can assume they are polynomials.

Suppose $P(y_0, z_0) = 0$, and the polynomial $P(y, z_0)$ has n distinct zeroes. Then we can define $y(z)$ near z_0, in such a way that $y(z_0) = y_0$.

Example: $y(z) = \sqrt{1 - z^2}$ is a solution to

$$y^2 + z^2 - 1 = 0.$$

This polynomial has distinct zeroes except when $z^2 = 1$, i.e., when $z = \pm 1$. We can define $y(z)$ around $z = 0$ by the Taylor series

$$1 - z^2/4 - z^4/8 - z^6/16 + \ldots$$

In general, there will be n "branches" (local expansions) corresponding to the n roots of $P(y, z) = 0$. Singularities of an algebraic function that are not poles are called *branch points* or *ramification points*.

Chomsky-Schützenberger Theorem:

I'll assume you are know a bit of formal language theory, in particular the ideas of regular language (something that is accepted by a finite automaton) and context-free language (something generated by a context-free grammar).

Let L be a set of strings over a finite alphabet. The *census function* of L is

$$C_L(n) = \#\{w \in L : |w| = n\}$$

Theorem:

If L is regular, then $f_L(n) = \sum_n C_L(n) z^n$ is rational.

If L is unambiguous context-free, then $f_L(n) = \sum_n C_L(n) z^n$ is algebraic.

Converse does not hold: if $L = a^n b^n c^n$, then the generating function for the census is $1/(1-3z)$, which is algebraic, even rational.

In many cases we can obtain f_L directly from a description of L (such as a grammar).

The result can be extended to general context-free languages, provided that we replace $C_L(n)$ by the sum, over all strings of length n in L, of the number of ways to derive that string.

Solving "regular" counting problems

Example: Let

$$L = \{x : \ x \text{ has no occurrence of } aa \ \},$$

where the alphabet is $\{a, b\}$.

We want to count how many strings of length n belong to L.

L is generated by the following grammar:

$$\begin{aligned} S &\to \ A \mid B \\ A &\to \ a \mid Ba \\ B &\to \ b \mid Sb \end{aligned}$$

How this works: Start with the symbols S. Until no capital letters remain, choose some capital letter and rewrite it using the above rules. Here is a sample derivation:

$$S \Rightarrow Ba \Rightarrow Sba \Rightarrow Aba \Rightarrow aba.$$

In words: A string in L can either end with an a or with a b. If it ends with an a, it must either be a itself or be something from L ending in b followed by a. If it ends in b, it is either b itself or some string in L, followed by b.

It can be shown that this grammar defines a regular set. (Why? The rewriting rules are always of the form $X \to Yz$, where Y is a symbol and z cannot be further rewritten.)

Furthermore, the grammar is unambiguous. This means that every string in the language can be generated in exactly one way.

Let $S(z)$, $A(z)$, and $B(z)$ be the generating functions for the counts of strings in L, strings in L ending with A, and strings in L ending in B, respectively. We get linear equations for these directly from the grammar:

$$\begin{aligned} S(z) &= A(z) + B(z) \\ A(z) &= z + zB(z) \\ B(z) &= z + zS(z) \end{aligned}$$

These equations are readily solved to obtain

$$\begin{aligned} S(z) &= \frac{2z + z^2}{1 - z - z^2} \\ A(z) &= \frac{z}{1 - z - z^2} \\ B(z) &= \frac{z + z^2}{1 - z - z^2} \end{aligned}$$

Expanding S in a Taylor series, we get

$$S(z) = 2z + 3z^2 + 5z^3 + 8z^4 + 13z^5 + \ldots,$$

so the solution to our counting problem is a Fibonacci number.

If you didn't recognize the sequence you could at this point use Darboux's method to figure out the asymptotics, as follows. The singularities of S are at

$$\alpha, \beta = \frac{-1 \pm \sqrt{5}}{2}.$$

Let β be the one nearest to the orgin, so that $\beta = (-1 + \sqrt{5})/2$. Near $z = \beta$, we have

$$S(z) \sim \frac{2\beta + \beta^2}{(\beta - \alpha)(\beta - z)} = \frac{\beta + 2}{(\beta - \alpha)(1 - z/\beta)} = \text{ const } (1 - z/\beta + z^2/\beta^2 + \cdots).$$

Since $\alpha\beta = -1$, this shows that

$$C_L(n) \sim \text{ const } \left(\frac{1 + \sqrt{5}}{2}\right)^n,$$

where the constant is

$$\frac{\beta + 2}{\beta - \alpha} = \frac{3\sqrt{5} + 5}{10} \doteq 1.170820393...$$

Solving "context-free" counting problems.

Defn: A binary (rooted, ordered, at most 2 children/node) tree is called *right-handed* if each node with a non-null left child has a non-null right child.

Here are the four right-handed trees with 4 nodes:

Let a_n be the number of right-handed binary trees with n nodes. We will estimate this by studying its generating function $R(z) = \sum_n a_n z^n$.

It's convenient to introduce "external nodes," which stand for the missing children.

Including the external nodes, we get:

Note that a tree with n nodes has $n+1$ external nodes. (You can prove that by induction.)

Then b_m, the number of right-handed trees with m external nodes, is the number of ways to generate x^m using the grammar

$$T \to TT \mid xT \mid xx$$

In words: a right-handed tree has either a) two subtrees, each of which is right-handed, or b) only the right subtree, which is right-handed, or c) no subtrees at all.

This grammar is ambiguous.

From this we see that the generating function

$$T(z) = \sum_m b_m z^m$$

satisfies the equation

$$T = T^2 + zT + z^2$$

This equation is nonlinear, so T is an algebraic function that isn't rational. (We get the nonlinear terms from the context-free productions that have more than one nonterminal on their right hand sides.)

Solving this equation, we have

$$T(z) = \frac{(1-z) - \sqrt{1-2z-3z^2}}{2} = z^2 + z^3 + 2z^4 + 4z^5 + 9z^6 + 21z^7 + \ldots$$

There is another branch with a plus sign, which we will ignore since its Taylor series coefficients are ultimately negative.

The coefficient of z^n is M_{n-2}, where M_n is the n-th *Motzkin number*. (For these numbers see R. Kemp, Fundamentals of the Average Case Analysis of Particular Algorithms, p. 77 ff.)

You should observe that this already gives us a better way to count the right-handed trees, since for $n \geq 2$, a_n is the twice the coefficient of z^{n+1} in the Maclaurin expansion of

$$\frac{\left(1-(2z+3z^2)\right)^{1/2}}{2},$$

which we can compute using the binomial theorem.

Middle Binomial Coefficients

It is known that

$$\log \binom{2\nu}{\nu} \sim \log \frac{4^\nu}{\sqrt{\pi\nu}} + \sum_{j\geq 1} \frac{C_j}{\nu^j},$$

where the constants C_j are related to Bernoulli numbers. [See, for example, Brent, Asymptotic approximation of central binomial coefficients with rigorous error bounds, 2016, arXiv:1608.048342.] In particular,

$$\log \binom{2\nu}{\nu} = \log \frac{4^\nu}{\sqrt{\pi\nu}} - \frac{1}{8\nu} + O(\nu^{-3}).$$

Exponentiating this, we get

$$\binom{2\nu}{\nu} = \frac{4^\nu}{\sqrt{\pi\nu}} \left(1 - \frac{1}{8\nu} + \frac{1}{128\nu^2} + O(\nu^{-3})\right) \tag{1}$$

The Leading Behavior of b_m

We will Darboux's method to estimate the Maclaurin series coefficients of

$$y(z) = -\sqrt{1-2z-3z^2}$$

Dividing by 2 then gives us what we want.

Lemma: We have

$$\binom{1/2}{m} = (-)^{m-1}\frac{1}{m}\binom{2m-2}{m-1}2^{-2m+1} = \frac{(-1)^{m-1}}{2\sqrt{\pi}m^{3/2}} + O\left(\frac{1}{m^{5/2}}\right)$$

This follows from (1).

Since

$$\sqrt{1-2z-3z^2} = \sqrt{(1+z)(1-3z)},$$

$y(z)$ is analytic ifor $|z| < 1/3$.

Note that for real variables y and z, the equation $y^2 = 1 - 2z - 3z^2$ defines an ellipse centered at $z = -1/3, y = 0$. The branch points are the places where dy/dz becomes infinite, that is, at $z = -1, 1/3$.

Near $z = 1/3$, we have

$$y(z) \sim -\frac{2}{\sqrt{3}}(1-3z)^{1/2}.$$

(More precisely, the two sides of this expression have a ratio that is analytic for $|z| < 1$, and equals 1 at $z = 1/3$.)

We can't get rid of the singularity at 1/3 by subtracting something but we can try to "improve" it somehow. Heuristically, we expect b_m, the m-th coefficient of $T(z)$ to be asymptotic to the m-th coefficient of $-\frac{1}{\sqrt{3}}\sqrt{1-3z}$, so

$$b_m = -\frac{1}{\sqrt{3}}\binom{1/2}{m}(-3)^m \sim \frac{3^{m-1/2}}{2\sqrt{\pi}m^{3/2}}.$$

(Here we have used the lemma.)

Taking $m = n+1$, we get an asymptotic formula for the number of right-handed binary trees with n nodes:

$$a_n \sim \frac{3^{n+1/2}}{2\sqrt{\pi}n^{3/2}}.$$

An Asymptotic Series for b_m

We use $w = \sqrt{1-3z}$ as a local parameter. Then z is a rational function of w, indeed, $z = (1-w^2)/3)$. Making this substitution we get

$$y = -\sqrt{(1-3z)(1+z)} = -\frac{2}{\sqrt{3}}w(1-w^2/4)^{1/2} = -\frac{2}{\sqrt{3}}w(1-w^2/8-w^4/128-\cdots)$$

(A series like this, in fractional powers, is called a *Puiseux expansion*.)

Applying (1), we get

$$\binom{1/2}{m} = \frac{(-1)^{m-1}}{2^{2m-1}m}\binom{2(m-1)}{m-1} = \frac{(-)^{m-1}}{2\sqrt{\pi}m^{3/2}}\left(1 + \frac{3}{8m} + \frac{25}{128m^2} + O(m^{-3})\right);$$

$$\binom{3/2}{m} = \frac{3/2}{m}\binom{1/2}{m-1} = (-)^{m-2}\frac{3}{4\sqrt{\pi}m^{5/2}}\left(1 + \frac{15}{8m} + O(m^{-2})\right);$$

$$\binom{5/2}{m} = \frac{(5/2)(3/2)}{m(m-1)}\binom{1/2}{m-2} = (-)^{m-3}\frac{15}{8\sqrt{\pi}m^{7/2}}\left(1 + O(m^{-1})\right).$$

From the expansion of y, we get (for z near 1/3)

$$T(z) \sim -\frac{(1-3z)^{1/2}}{\sqrt{3}} + \frac{(1-3z)^{3/2}}{8\sqrt{3}} + \frac{(1-3z)^{5/2}}{128\sqrt{3}}.$$

After expanding powers of $1-3z$ by the binomial theorem and using our asymptotic formulas for the binomial coefficients, we get

$$b_m = \frac{3^{m-1/2}}{2\sqrt{\pi}m^{3/2}}\left(1+\frac{9}{16m}+\frac{265}{512m^2}+O(m^{-3})\right)$$

Values of a_n can be obtained by substituting $m = n+1$ into this formula.

You can see how accurate this is by comparing 835, the actual number of right-handed trees for $n = 10$, to 834.645..., computed from the three-term approximation above. The relative error is about 4 parts in 10,000. From this we can surmise that the implied constant in the big-O term is probably less than 1.

A Rigorous Error Bound

So far the derivation was heuristic. With more work, we can prove that the first approximation for b_m has the error claimed.

Recall that

$$w = \sqrt{1-3z},$$

and set $\hat{y} = -2w/\sqrt{3}$. We have

$$y(z) - \hat{y}(z) = w^3 h(z),$$

where

$$h(z) = \frac{(1+z)^{1/2} - 2/\sqrt{3}}{1-3z} = \sum_{j\geq 0} c_j z^j$$

is analytic for $|z| < 1$.

The m-th Maclaurin series coefficient of $y - \hat{y}$ is

$$\sum_{j=0}^{m} (-3)^j \binom{3/2}{j} c_{m-j},$$

which we estimate as follows. We must have

$$|c_j| = O\left((3/2)^j\right),$$

since h is analytic at least out to $|z| = 2/3$. Also, by our lemma

$$\binom{3/2}{j} = \frac{3/2}{j}\binom{1/2}{j-1} = O(j^{-5/2}).$$

Therefore

$$\left|\sum_{j=0}^{\lfloor m/2\rfloor} (-3)^j \binom{3/2}{j} c_{m-j}\right| = O\left(3^{m/2}(3/2)^m)\right) = O\left((3^{3/2}/2)^m\right)$$

(since $\sum_j j^{-3/2}$ converges), and

$$\left| \sum_{j=\lceil m/2 \rceil}^{m} (-3)^j \binom{3/2}{j} c_{m-j} \right| = O\left(\frac{3^m}{m^{5/2}} \sum_{k=0}^{\lfloor m/2 \rfloor} 2^{-k} \right) = O\left(\frac{3^m}{m^{5/2}} \right)$$

Therefore,

$$\sum_{j=0}^{m} (-3)^j \binom{3/2}{j} c_{m-j} = O\left(\frac{3^m}{m^{5/2}} \right),$$

so

$$b_m = \frac{3^{m-1/2}}{2\sqrt{\pi} m^{3/2}} \left(1 + O(m^{-1})\right).$$

Notes.

Every regular language has an unambiguous left-linear grammar. See, e.g. M. A. Harrison, Introduction to Formal Language Theory, Section 2.5, where the result is proved for right-linear grammars.

By multiplying the Maclaurin series for $(1+z)^{1/2}$ and $(1-3z)^{1/2}$, we can get an explicit formula for the number of right-handed binary trees with m external nodes. It is (for $m \geq 2$)

$$\sum_{\ell=0}^{m} \binom{1/2}{\ell} \binom{1/2}{m-\ell} (-3)^\ell.$$

This looks like a hypergeometric sum.

Balanced parenthesis strings also make a good example.

The "textbook" grammar for this language is

$$S \longrightarrow \epsilon | SS | (S) \ .$$

Unfortunately this is highly ambiguous: every string in the language has infinitely many derivations.

A better choice is the unambiguous grammar (suggested by Samuel Jackson)

$$S \longrightarrow \epsilon | (S)S \ .$$

Using this grammar, counting works as expected. The generating function for the number of length n balanced parenthesis strings satisfies

$$S = 1 + z^2 S^2,$$

so

$$S = \frac{1 - \sqrt{1-4z^2}}{2z^2} = 1 + z^2 + 2z^4 + 5z^6 + 14z^8 + 42z^{10} + \cdots \ .$$

(Taking the minus sign makes GF coefficients positive.) The coefficients are called Catalan numbers.

As an exercise you can consider the grammar

$$S \longrightarrow ()|SS|(S) \ ,$$

which generates all positive length balanced parenthesis strings. The total number of derivations for strings of length n has a generating function satisfying

$$S = z^2 + S^2 + z^2 S.$$

A linear CF grammar is one with at most one variable on each right hand side. Languages with such grammars will also have rational census generating functions, even if they are not regular. One example is the even length palindromes made of 0's and 1's, which has the unambiguous grammar

$$S \longrightarrow \epsilon|0S0|1S1.$$

If L is this language, then

$$C_L(n) = \begin{cases} 2^{n/2}, & n \text{ even;} \\ 0, & n \text{ odd.} \end{cases}$$

So the generating function for $C_L(n)$ is $(1 - 2z^2)^{-1}$.

Lecture 24

Topic du jour: What do to when Darboux fails; using the residue theorem.

References: GK 4.3.2.

Review of Darboux method

Goal: estimate a_n asymptotically

Find singularities of $f(z) = \sum_n a_n z^n$ that are nearest to the orgin.

Find a simpler function $g(z) = \sum_n b_n z^n$ that matches these singularities (in some sense).

Conclude that $a_n \sim b_n$.

Entire functions

Defn. f is an *entire function* if it is analytic over the entire complex plane.

Examples: polynomials, $\exp(z)$, $\exp(z^2)$, etc.

When a sequence has an entire generating function, Darboux will not work since there are no singularities.

Alternate method: use the residue theorem.

Recall that

$$\frac{1}{2\pi i}\int_{|z|=R} z^m dz = \begin{cases} 0, & \text{if } m \neq -1; \\ 1, & \text{if } m = -1. \end{cases}$$

To estimate a_n, we form $f(z) = \sum_{n\geq 0} a_n z^n$. Then

$$\frac{f(z)}{z^{n+1}} = \frac{a_0}{z^{n+1}} + \cdots + \frac{a_n}{z} + \cdots,$$

and if we are allowed to interchange summation and integration we find

$$\frac{1}{2\pi i}\int_{|z|=R} \frac{f(z)dz}{z^{n+1}} = \frac{a_n}{2\pi i}\int_{|z|=R} \frac{dz}{z} = a_n.$$

This expresses the n-th element of our sequence as an integral, which we can then try to estimate.

Example 1: Involutions

Defn: An *involution* is a permutation σ for which $\sigma^2 = 1$.

Alternate definition: An involution is a permutation composed only of fixed points and 2-cycles.

Intuitively, we expect involutions to be rare. (Why?) We would like to state this idea precisely.

Let I_n be the number of involutions on $\{1, \ldots, n\}$. Then

$$I_n = n! \sum_{k+2\ell=n} \frac{1}{k! 2^\ell \ell!}$$

Why is this true? Consider an involution with k 1-cycles and ℓ 2-cycles. (We have to have $k + 2\ell = n$.) It must look like this:

$$(*)(*)\ldots(*)(**)(**)\cdots(**).$$

Suppose the entries are filled in somehow. How many other ways will give us the same permutation? We can rearrange the 1-cycles in $k!$ ways, and the 2-cycles in $\ell!$ ways, as well as permute the individual entries in 2^ℓ ways. Therefore, there are $n!/(k!\ell!2^\ell)$ permutations with this cycle structure. Summing over the possible values of k and ℓ gives the result.

On the other hand,

$$e^{z+z^2/2} = \left(\sum_{k\geq 0} \frac{z^k}{k!}\right)\left(\sum_{\ell \geq 0} \frac{z^{2\ell}}{2^\ell \ell!}\right) = \sum_{n\geq 0} \frac{I_n}{n!} z^n.$$

We say that $\exp(z^2/2 + z)$ is the *exponential generating function* for I_n.

Estimating $I_n/n!$ using the integral.

If $w = x + iy$ with x, y real, then

$$|e^w| = e^x.$$

Therefore

$$|e^{z^2/2+z}| \leq e^{|z|^2/2+|z|}.$$

Since I_n is positive, we have (setting $z = Re^{i\theta}$)

$$\begin{aligned} \frac{I_n}{n!} &= \left| \frac{1}{2\pi i} \int_{|z|=R} \frac{e^{z^2/2+z} dz}{z^{n+1}} \right| \\ &\leq \frac{1}{2\pi} \int_{-\pi}^{\pi} \frac{e^{R^2/2+R} d\theta}{R^n} \\ &= \frac{e^{R^2/2+R}}{R^n}. \end{aligned}$$

Crucially, this is valid for every $R > 0$.

Call this estimate $g(R)$. We note that g blows up as $R \to 0$ and $R \to \infty$. Thus, g will have a minimum value which we can find by differentiation. We have

$$\log g = R^2/2 + R - n \log R$$

and

$$\frac{d}{dR} \log g = R + 1 - n/R;$$

this equals zero when

$$R = \frac{-1 + \sqrt{1 + 4n}}{2} \sim \sqrt{n}.$$

So $R = \sqrt{n}$ should give us a pretty good estimate. We find

$$\frac{I_n}{n!} \leq \frac{e^{n/2+\sqrt{n}}}{n^{n/2}},$$

and by applying Stirling's formula,

$$I_n \leq e^{n/2 \log n - n/2 + \sqrt{n} + O(\log n)}.$$

Dividing this by

$$n! = e^{n \log n - n + O(\log n)},$$

we get

$$I_n \leq e^{n/2 \log n + O(n)}.$$

Very roughly, then, the number of inversions is no more than the square root of the number of permutations.

Here is a simple combinatorial argument to show that the lead term in our upper bound is correct. Suppose we take $\ell = \lfloor n/2 \rfloor$ and $k = 0, 1$ as appropriate. Then, since a sum of positive numbers is lower bounded by any term,

$$I_n \geq \frac{n!}{\lfloor n/2 \rfloor!} = e^{n/2 \log n + O(n)}.$$

Hayman's theorem (J. Reine Angew. Math. 1956; see also Harris and Schoenfeld, Ill. J. Math. 1968):

Our method only gives estimates for generating function coefficients. Here is a result that gives the asymptotics.

Let p be a polynomial with real coefficients, $f(z) = \sum_n a_n z^n$ where a_n is positive for all large n. Further, let $w(x) = xf'(x)/f(x)$, and $b(x) = xw'(x)$. If for all large n, $w(x) = n$ has a unique positive root r_n, then

$$a_n \sim \frac{f(r_n)}{r_n^n} \cdot \frac{1}{\sqrt{2\pi b(r_n)}}.$$

Think of this as our estimate, times a correction factor.

Applied to involutions, we get

$$I_n = e^{n/2 \log n - n/2 + \sqrt{n} + O(1)}$$

Section 5.4 of Wilf's book, Generatingfunctionology, has more details on using Hayman's theorem to count involutions.

Notes

The following theorem may be of interest. Suppose

$$f(z) = \sum_{n \geq 0} a_n z^n$$

is an entire function. Let $M(r)$ be the maximum of $|f|$ on the circle of radius r. Define

$$\nu = \limsup \frac{\log M(r)}{r}$$

and

$$\mu = -\limsup \frac{\log |a_n|}{n \log n}.$$

Then $\mu = 1/\nu$. (See e.g. Hardy, Orders of Infinity, p. 52.) As an example, we have the exponential function for which $\mu = \nu = 1$. You can find more results of this type in Polya-Szego, Problems and Theorems in Analysis II, pp. 3-14.

To estimate power series coefficients one can also use bounds on the real part and the value at the center. See Satz 225 of E. Landau, Vorlesungen u. Zahlentheorie.

Lecture 25

Topic du jour: Approximations for "powerful" integrals

References: Wilf, Mathematics for the Physical Sciences (for Laplace approximation); L. Sirovich, Techniques of Asymptotic Analysis (for Watson's lemma). The birthday problem analysis follows B. Arnold, AMM Nov. 1971, pp. 1022 ff.

What's this about?

Last time we showed how to express the n-th coefficient of a power series as an integral. In particular, if $f(z) = \sum a_n z^n$ then

$$a_n = \frac{1}{2\pi i} \int \frac{f(z)}{z^{n+1}} dz.$$

The main difficulty in evaluating such an integral is the high power in the integrand. Integrals with this type, of the form

$$\int f(x)^n g(x) dx$$

can often be evaluated asymptotically.

Will discuss two methods.

The Laplace approximation

Watson's lemma

An integral for a combinatorial sum

Let

$$S_n = \sum_{i=0}^{n} \binom{n}{i}^2 2^i \sim ???.$$

We note that

$$\begin{aligned} S_n &= \text{ constant term of } (1+x)^n (2+1/x)^n \\ &= \text{ coefficient of } x^{-1} \text{ in } (2x+3+1/x)^n / x \\ &= \frac{1}{2\pi i} \int_{|x|=R} \frac{(2x+3+1/x)^n dx}{x} \end{aligned}$$

We can choose any R we want. What is a good choice?

Let $f(x) = 2x + 3 + x^{-1}$, so that our integral comes from f raised to some humongous power.

We will choose R so that the circle $|z| = R$ contains a zero P of f'.

The reasons for this are twofold. First, P is a critical point for $|f|^2$. We can see this as follows. Suppose along the path of integration, $f = u(t) + iv(t)$. Then

$$\frac{d}{dt}|f|^2 = 2u\frac{du}{dt} + 2v\frac{dv}{dt} = 0.$$

Our hope is that P actually maximizes $|f|^2$ along the path, which is the same as maximizing $|f|$. Then the integral will be concentrated near P, and we can replace the integrand by something simpler.

Note that by the maximum principle, we can only hope to maximize $|f|$ along some path. We will not maximize it at all points nearby P, so that P will be (if we are lucky) a saddle point of $|f|$.

We get a second benefit from our choice of R. Since we are approaching the critical point along a curve for which $|z|$ is constant,

$$\frac{d}{dz}\arg f = \frac{d}{dz}\log f = \frac{f'}{f} = 0.$$

This means that we are essentially dealing with a real integral.

For our f, we find that we want $R = 1/\sqrt{2}$.

Making the substitution $x = e^{i\theta}/\sqrt{2}$ in the integral, we get

$$S_n = \frac{1}{2\pi}\int_{-\pi}^{\pi}(3 + 2\sqrt{2}\cos\theta)^n d\theta$$

As promised, the integrand has a maximum at $\theta = 0$ and dies away rapidly as $\theta \to \pm\pi$.

Note that $3 + 2\sqrt{2}\cos\theta$ is about 5.8 when $\theta = 0$, and 0.17 when $\theta = \pm\pi$.

Laplace approximation

This is a standard trick for getting the asymptotics of $\int_a^b f(\theta)^n d\theta$, when f is a smooth function with a unique maximum in the interval of integration. The method works more or less well depending on how much of the integrand is concentrated around the maximum point.

We'll proceed heuristically. The idea is to replace the integrand with a constant times a Gaussian (normal density) function that matches up to second order terms, then integrate over all θ.

This is the old, more or less reliable, "method of moments." We will be approximating the integrand by

$$Ce^{(\theta-\mu)^2/(2\sigma^2)},$$

so there are three parameters to play with. Thus, we will match the first three Taylor coefficients of the integrand.

We have (as $\theta \to 0$)

$$3 + 2\sqrt{2}\cos\theta \sim (3 + 2\sqrt{2})\left(1 - \frac{\sqrt{2}}{3 + 2\sqrt{2}}\theta^2\right),$$

so

$$(3 + 2\sqrt{2}\cos\theta)^n \sim (3 + 2\sqrt{2})^n \left(1 - \frac{\sqrt{2}n}{3 + 2\sqrt{2}}\theta^2\right) \sim (3 + 2\sqrt{2})^n \exp(-\theta^2/(2\sigma^2)),$$

provided that

$$\sigma^2 = \frac{3 + 2\sqrt{2}}{2\sqrt{2}n}.$$

We guess, then, that

$$S_n \sim \frac{1}{2\pi}\int_{-\infty}^{\infty}(3 + 2\sqrt{2})^n e^{-\theta^2/(2\sigma^2)} = \frac{1}{2\pi}(3 + 2\sqrt{2})^n\sqrt{2\pi}\sigma = \frac{(3 + 2\sqrt{2})^{n+1/2}}{2^{5/4}\sqrt{\pi n}}.$$

The following table shows the quality of this approximation.

n	exact	approx	ratio
1	3	3.34	1.1126
2	13	13.76	1.0582
5	1683	1722.60	1.0235
10	8097453	8192698.28	1.0118
20	260543813797441	262075045635766.44	1.0059

Using ideas of H. S. Wilf (Mathematics for the Physical Sciences, pp. 127-131) we give a criterion for the Laplace approximation to be valid. Suppose that h is a real function satisfying the following three conditions. First

$$\int_{-\infty}^{\infty} e^{nh(x)}dx$$

exists for all sufficiently large n. Second, there is a constant σ such that

$$h(x) \sim -\frac{x^2}{2\sigma^2}$$

as $x \to 0$. Third, $h \le 0$, and it is bounded away from 0 on every set $|x| \ge \delta$. Then

$$\int_{-\infty}^{\infty} e^{nh(x)}dx \sim \frac{\sqrt{2\pi}\sigma}{n^{1/2}}.$$

The birthday problem

We draw uniform random samples (with replacement) from $\{1, \ldots, n\}$ until some sample appears twice.

Surprising fact: after $\Theta(\sqrt{n})$ samples, there is probably a duplicate.

This result is useful in the study of hashing, in integer factoring, and in cryptanalysis. (More about that later.)

A crude analysis

What can we conclude about this using what we already know?

By the pigeonhole principle, there will surely be a duplicate after $n+1$ samples. Even if there are αn samples and $\alpha < 1$, our work on occupancy tells us that the expected number of distinct samples is asymptotic to

$$n(1 - e^{-\alpha}) < n\alpha.$$

The following inequalities hold for events $E_1, \ldots, E_r$ in any probability space.

$$\sum_i \Pr[E_i] - \sum_{i<j} \Pr[E_i \cap E_j] \leq \Pr[\exists i E_i] \leq \sum_i \Pr[E_i].$$

Suppose there are m samples. For $i = 1, \ldots, \binom{m}{2}$, let us take E_i to be the event that the i-th pair of samples match. We have

$$\Pr[E_i] = 1/n$$

whereas if $i \neq j$,

$$\Pr[E_i \cap E_j] = 1/n^2$$

(this is true whether the pairs overlap or not).

Thus

$$\Pr[\text{ there is a duplicate in } m \text{ samples }] \leq \binom{m}{2}\frac{1}{n} \sim \frac{m^2}{2n},$$

which tells us we should not expect to find a duplicate if there are $o(\sqrt{n})$ samples.

On the other hand,

$$\Pr[\text{ there is a duplicate in } m \text{ samples }] \geq \binom{m}{2}\frac{1}{n} - \binom{\binom{m}{2}}{2}\frac{1}{n^2} \sim \frac{m^2}{2n} - \frac{m^4}{8n^2},$$

and the right hand side is significant ($= 3/8$) when $m = \sqrt{n}$.

This suggests we should try to prove that the expected time when a duplicate occurs is $\Theta(\sqrt{n})$.

An integral we'll need

Recall the *Euler gamma integral*:

$$\Gamma(i+1) = i! = \int_0^\infty e^{-x} x^i dx.$$

From this we get the Laplace transform of x^i:

$$\frac{i!}{n^{i+1}} \int_0^\infty e^{-nx} x^i dx.$$

We will also need the special values

$$\Gamma(1/2) = (-1/2)! = \sqrt{\pi}$$

and

$$\Gamma(3/2) = (1/2)! = \sqrt{\pi}/2.$$

Proof:

$$\int_0^\infty x^{1/2} e^{-x} dx = \frac{1}{\sqrt{2}} \int_0^\infty t^2 e^{-t^2/2} dt = \sqrt{\pi} \times \frac{\text{variance of standard normal}}{2},$$

So $(1/2)! = \sqrt{\pi}/2$. Also, $(-1/2)! = 2(1/2)! = \sqrt{\pi}$.

Asymptotic evaluation of the mean time for duplication.

Let X be the number of samples needed to find a duplicate. Then

$$\Pr[X > i] = 1(1-1/n)(1-2/n)\cdots(1-(i-1)/n) = \frac{n^{\underline{i}}}{n^i}$$

Note that this vanishes when $i \geq n+1$, as it should.

Therefore

$$\begin{aligned}
E[X] &= \sum_{i=0}^{n} \Pr[X > i] \\
&= n \sum_{i=0}^{n} \binom{n}{i} \frac{i!}{n^{i+1}}. \\
&= n \sum_{i=0}^{n} \binom{n}{i} \int_0^\infty e^{-nx} x^i dx \\
&= n \int_0^\infty e^{-nx} \sum_{i=0}^{n} \binom{n}{i} x^i dx \\
&= n \int_0^\infty e^{-nx} (1+x)^n dx.
\end{aligned}$$

(Here we have used our Laplace transform formula and the binomial theorem.)

This integral can be evaluated asymptotically using the Laplace approximation.

Note that as $x \to 0$,

$$e^{-x}(1+x) \sim (1 - x + x^2/2)(1+x) = 1 - x^2/2 \sim e^{-x^2/2}$$

Therefore

$$E[X] = n \int_0^\infty \{e^{-x}(1+x)\}^n dx \sim \frac{n}{2} \int_{-\infty}^\infty e^{-nx^2/2} dx = \sqrt{\frac{\pi n}{2}}.$$

Caveat: Again, we didn't prove this rigorously.

As an example, this formula gives for $n = 365$ the approximation 23.9445... .

Watson's lemma

To get a better approximation to $E[X]$, we can use a trick due to Watson, which gives the asymptotics for integrals of the form

$$\int_0^\infty e^{-nu} f(u) du$$

This isn't exactly our integral but we'll cast it into that form shortly.

Watson's lemma: Let $-1 < \alpha_0 < \alpha_1 < ...$ be numbers such that

$$f(u) \sim a_0 u^{\alpha_0} + a_1 u^{\alpha_1} + a_2 u^{\alpha_2} + \cdots$$

as $u \to 0^+$. Assume that f is well-behaved enough to ensure the existence of the Laplace transform $\int_0^\infty e^{-nu} f(u) du$ for large n. (In particular, we require $f(u) = O(e^{bu})$.) Then

$$\int_0^\infty e^{-nu} f(u) dt \sim a_0 \frac{\alpha_0!}{n^{\alpha_0+1}} + a_1 \frac{\alpha_1!}{n^{\alpha_1+1}} + a_2 \frac{\alpha_2!}{n^{\alpha_2+1}} + \cdots$$

For a proof of this, see Sirovich, p. 66.

The basic idea behind this lemma is that when n is large e^{-nu} decreases very sharply, so we can approximate the integral by plugging in the asymptotic expansion and integrating term-by-term.

Formally we have

$$\int_0^\infty e^{-nu} f(u) du = \int_0^\infty e^{-nu} \sum_i a_i u^{\alpha_i} du = \sum_i a_i \int_0^\infty e^{-nt} u^{\alpha_i} du = \sum_i a_i \frac{\alpha_i!}{n^{\alpha_i+1}}.$$

Better approximations for the duplication time

Observe that

$$\frac{E[X]}{n} = \int_0^\infty \{e^{-x}(1+x)\}^n dx = \int_0^\infty e^{-n(x-\log(1+x))} dx.$$

Substitute $u = x - \log(1+x)$, and this becomes

$$\frac{E[X]}{n} = \int_0^\infty e^{-nu} \frac{1+x(u)}{x(u)} du$$

We now need to find an asymptotic expansion for $x(u)$ near $u = 0$.

We have

$$u = x - \log(1+x) = x^2/2 - x^3/3 + x^4/4 - \cdots$$

so

$$2u = x^2(1 - (2/3)x + (1/2)x^2 - (2/5)x^3 + \cdots),$$

and

$$x = \frac{\sqrt{2u}}{(1 - (2/3)x + (1/2)x^2 - (2/5)x^3 + \cdots)^{1/2}}$$

We can get asymptotic approximations to x by starting with $x_0 = \sqrt{2u}$ and repeatedly plugging into this equation. For example, we have

$$\begin{aligned} x = &\ \sqrt{2u}(1 - \frac{2}{3}\sqrt{2u} + O(u))^{1/2} \\ = &\ \sqrt{2u} + (2/3)u + O(u^{3/2}), \end{aligned}$$

so we can take $x_1 = \sqrt{2u} + (2/3)u$.

Repeating this process, we find

$$x = \sqrt{2}u^{1/2} + (2/3)u + (\sqrt{2}/18)u^{3/2} - \frac{2}{135}u^2 + \cdots$$

and

$$\frac{1+x}{x} = (\sqrt{2}/2)u^{-1/2} + (2/3)u + (\sqrt{2}/12)u^{1/2} + \frac{4}{135}u + \cdots$$

Applying Watson's lemma,

$$\begin{aligned} \frac{E[X]}{n} = &\ \int_0^\infty e^{-nu} \frac{1+x(u)}{x(u)} du \\ = &= (\sqrt{2}/2)\frac{(-1/2)!}{n^{1/2}} + (2/3)\frac{0!}{n} + (\sqrt{2}/12)\frac{(1/2)!}{n^{3/2}} + \frac{4}{135}\frac{1!}{n^2} + \cdots \end{aligned}$$

so

$$E[X] = \sqrt{\frac{\pi n}{2}} + \frac{2}{3} + \frac{1}{12}\sqrt{\frac{\pi}{2n}} + \frac{4}{135n} + \cdots$$

The relative error in this approximation is $O(n^{-5/2})$.

When $n = 365$, the first 4 terms give

$$E[X] = 24.61659...$$

which is accurate to 6 figures.

More on the birthday problem

There is an elementary argument (see, e.g. Feller, vol. 1) to show that the median of X is also $\Theta(\sqrt{n})$.

The random variable $X/\sqrt{n}$ has an asymptotic distribution:

$$\Pr[X < \alpha\sqrt{n}] \sim \int_0^\alpha x e^{\frac{-x^2}{2}} dx.$$

[B. Harris, Ann. Math. Stat. 31, 1960].

This is the *Rayleigh* distribution.

It is the distribution of the absolute value of a standard normal random variable.

From this result we can compute

$$\sigma^2(X) \sim (2 - \pi/2)n.$$

The integral for $E[X]$ can also derived using the Poisson process and Wald's equation. (Similar to the integral for coupon collection.) This approach gives a way to compute the expectation numerically, when the probabilities are non-uniform.

For applications to factoring, see J. M. Pollard, BIT, v. 15, 1975, pp. 331-334.

Our estimates for $E[X]$ have the following interesting consequence: if you have a hash table for storing n keys and wish to make collisions unlikely, you must the table very large, indeed, greater than n^2 in size. Most "practical" hash tables have size proportional to n and therefore must handle collisions.

A related problem has applications to cryptography: Suppose we choose two groups of random samples, say $X_1, \ldots, X_m$ and $Y_1, \ldots, Y_m$, from $\{1, \ldots, m\}$. What is the chance that one of the X_i's equals one of the Y_j's? See K. Nishimura and M. Sibuya, "Occupancy with two types of balls," Ann. Inst. Statist. Math., v. 40, 1988, pp. 77-91.

Notes

The asymptotic expansion for the expected waiting time in the birthday problem is from R. J. Dickson, Solution to Problem E 2263, AMM 1971, pp. 1023-1024.

> Using martingales and optional stopping it is easy to show that $E[T(T-1)] = 2N$ (second factorial moment). This then gives a series for the variance (or standard deviation) of T in the birthday problem.

Expanding into an integral and interchanging summation and integration is a trick that works in other cases. For example, let's compute $E[1/(X+2)]$ when X has a Poisson distribution. We observe that

$$\begin{aligned}
\sum_{k\geq 0} \frac{\lambda^k}{k!(k+2)} &= \lambda^{-2} \sum_{k\geq 0} \frac{\lambda^{k+2}}{k!(k+2)} \\
&= \lambda^{-2} \sum_{k\geq 0} \frac{\int_0^\lambda t^{k+1} dt}{k!} \\
&= \lambda^{-2} \int_0^\lambda t e^t dt \\
&= \lambda^{-2} \left[\lambda e^\lambda - e^\lambda + 1\right]
\end{aligned}$$

Therefore,

$$E[1/(X+2)] = \frac{\lambda - 1 + e^{-\lambda}}{\lambda^2}$$

(The problem is from forthcoming work by D. Eager and M. Vernon)

Another approach to two-ball occupancy: Stone and Thiebaut, "Footprints in the Cache," Proc. Performance 86.

We can use the birthday problem to model a random nonlinear shift register. The idea is to look at i.i.d. random bits until some k-bit pattern repeats. See E. D. Karnin, The first repetition of a pattern in a symmetric Bernoulli sequence, J. Appl. Prob. 20, 1983, 413–418.

The method of stationary phase could be discussed here. (Reference: Bender/Orszag p. 276 ff.) This finds the asymptotics for an oscillatory integral using Watson's lemma.

Max-to-sum ratios make a nice application for the Laplace approximation.

Lecture 26

Topic du jour: Models for random binary trees

References: Purdom and Brown 7.1.1.1 (for random insertion trees). Knuth I, 2.3.4.5 ex.$\tilde{5}$ (for uniform trees).

Binary trees

We'll agree that a binary tree is one in which each node has 0,1, or 2 children. We distinguish between left and right children.

How many binary trees are there?

Let t_n be the number of binary trees with n nodes. Starting with $t_0 = 1$, the sequence goes

$$1, 1, 2, 5, 14, 42, 132, 429, 1430, 4862, \ldots$$

The generating function for t_n is

$$G(w) := \sum_{n \geq 0} t_n w^n = \frac{1 - \sqrt{1 - 4w}}{2w}.$$

This follows from some results we will prove below. Alternatively, you can use the "context-free" counting ideas we have developed.

From the binomial theorem we get the closed form:

$$t_n = \frac{1}{n+1} \binom{2n}{n}.$$

These are called the *Catalan numbers*.

Using Stirling's formula,

$$t_n \sim \frac{4^n}{\sqrt{\pi n^3}}.$$

Models of random binary trees

Uniform: Each tree with n nodes is equally likely.

Random insertion: Use n keys (in random order) to determine a search tree. (The first key is the root, the second becomes the left child if it is smaller than the root, the right child if larger, and so on.)

These models behave very differently.

The uniform model produces trees of depth $\Theta(\sqrt{n})$.

The random insertion model produces trees of depth $\Theta(\log n)$.

We will prove this for a measure of depth called the *internal path length*.

If T is a binary tree, then

$$I(T) = \sum_x \text{dist}(x, \text{root}).$$

Thus $I(T)/n$ is the depth of a random node in the tree.

Similar results also hold for the maximum path length but are harder to prove.

Analysis of internal path length under random insertion distribution

Let $x_1, \ldots, x_n$ be a random permutation of $1, \ldots, n$.

Build the tree recursively. The root is x_1, and the left (resp. right) subtree is formed from the keys that are less than (resp. greater than) x_1.

We want to know the expected value of $I(T)$ in this model.

Condition on ℓ, the number of nodes in the left subtree.

Note that $0 \leq \ell \leq n-1$, with a uniform distribution.

We have

$$\begin{aligned} E_n := E[I(T)] &= (n-1) + E[I(L)] + E[I(R)] \\ &= (n-1) + 2E[I(L)] \\ &= (n-1) + 2\sum_{\ell=0}^{n-1} E[I(L)||L| = \ell] \Pr[|L| = \ell]. \end{aligned}$$

Therefore

$$E_n = (n-1) + \frac{2}{n}\sum_{\ell=0}^{n-1} E_\ell,$$

with $E_0 = E_1 = 0$.

We have solved this equation before. From our work on quicksort it follows that

$$E_n = 2n \ln n + O(n).$$

This, the expected height of a tree (under the random isertion model) is $2 \ln n + O(1)$.

Analysis of internal path length under uniform distribution

Theorem: For the uniform distribution,

$$E\left[\frac{I(T)}{n}\right] \sim \sqrt{\pi n}.$$

To prove this we consider the *structural generating function*

$$Q(w,z) := \sum_T w^{|T|} z^{I(T)}.$$

The sum is taken over all trees. (We write $|T|$ for the number of nodes in T.)

Each nonempty tree is a root, together with two subtrees L and R, so

$$\begin{aligned} Q(w,z) &= 1 + \sum_{L,R} w^{|L|+|R|+1} z^{I(L)+|L|+I(R)+|R|} \\ &= 1 + w \left(\sum_L w^{|L|} z^{I(L)+|L|} \right) \left(\sum_R w^{|R|} z^{I(R)+|R|} \right) \end{aligned}$$

Therefore,

$$Q(w,z) = wQ(wz,z)^2 + 1.$$

Aside: If we plug in $z = 1$ and solve for Q, we get the generating function for t_n.

Observe that

$$\frac{\partial Q}{\partial z} = \sum_T I(T) w^{|T|} z^{I(T)-1},$$

so

$$q(w) := \frac{\partial Q}{\partial z}(w,1) = \sum_T I(T) w^{|T|}$$

is the generating function for

$$A_n = \sum_{|T|=n} I(T).$$

We now compute $q(w)$.

From the relation $Q(w,z) = wQ(wz,z)^2 + 1$ we get

$$\begin{aligned} \frac{\partial Q}{\partial z} &= 2wQ(wz,z) \frac{\partial}{\partial z} Q(wz,z) \\ &= 2wQ(wz,z) \left[wQ_1(wz,z) + Q_2(wz,z) \right] \end{aligned}$$

Now plug in $z = 1$ and note that $G(w) = Q(w,1)$. We get

$$q(w) = 2wG(w)[wG'(w) = q(w)].$$

This is a linear equation for q which we can solve to get

$$q(w) = \frac{2w^2 GG'}{1 - 2wG},$$

where

$$G(w) = \frac{1 - \sqrt{1-4w}}{2w}.$$

Note that q is an algebraic function. To understand it, we change variables.

Introduce the new variable $t = \sqrt{1-4w}$, so that $w = (1-t^2)/4$.

Then

$$G = \frac{1-t}{(1-t^2)/2} = \frac{2}{1+t}$$

and

$$\frac{dG}{dw} = \frac{dG}{dt}\frac{dt}{dw} = \frac{4}{(1+t)^2 t}.$$

Substituting these into q, we get

$$q = \frac{(t-1)^2}{t^2(1+t)} = \frac{1}{t^2} - \frac{3}{t} + 4 - 4t + \cdots = \frac{1}{1-4w} - \frac{3}{\sqrt{1-4w}} + 4 - 4\sqrt{1-4w} + \cdots$$

The main term of q is

$$\frac{1}{t^2} = \frac{1}{1-4w} = 1 + 4w + \cdots + 4^n w^n + \cdots$$

so let us work with

$$B_n = A_n - 4^n.$$

Its generating function is

$$\sum_{n \geq 0} B_n w^n = -\frac{3}{\sqrt{1-4w}} + [\text{ analytic function of } \sqrt{1-4w}\ ],$$

so we can apply Darboux to get $B_n = O(4^n/\sqrt{n})$. This implies

$$A_n \sim 4^n.$$

An exact formula for A_n is known. We have

$$A_n = 4^n - \frac{3n+1}{n+1}\binom{2n}{n}.$$

See Chottin and Cori, RAIRO Informatique Théorique 1982.

This gives

$$E[I(T)] = \frac{A_n}{\binom{2n}{n}/(n+1)} \sim \frac{4^n}{4^n/\sqrt{\pi n^3}} \sim \sqrt{\pi} n^{3/2}.$$

so

$$E\left[\frac{I(T)}{n}\right] \sim \sqrt{\pi n}.$$

Results on height

As another measure of tree depth we can consider

$$M(T) = \max_x\{\#\text{ of edges between root and } x\}.$$

If T is drawn from the random insertion distribution on n-node binary trees, then

$$E[M(T)] \sim c \ln n,$$

where $c \geq 2$ and

$$c \log(2e/c) = 1.$$

We have $c \doteq 4.31107$.

This was proved by Robson, Australian Computer Journal, v.11, 1969. (See also Devroye, J. ACM 33, 1986).

Flajolet and Odlyzko (JCSS 25, 1982) showed that for the uniform distribution,

$$E[M(T)] \sim 2\sqrt{\pi n}.$$

Notes

A great reference on random search trees is Mahmoud's book, The Evolution of Random Search Trees.

There are a bunch of other random tree models.

Galton-Watson branching processes, in which each node (organism) can have a random number of children. Restricting this to be ≤ 2, we get binary trees.

Random code trees (as generated by, say, the Lempel-Ziv data compression algorithm).

Random recursive trees. See R. T. Smythe and H. M. Mahmoud, Theor. Prob. Math. Stat. v. 51, 1995.

Branching random walks. Gives a proof of the Robson theorem. Nice short paper on this by Biggins and Grey, Stat. Prob. Lett., 1997.

It may be incorrect to speak of *the* average number of key comparisons done by Quicksort, since the precise number can depend on implementation. In particular, there are Quicksort variants with $n+1$ (not $n-1$) comparisons at the top level. See, e.g. Knuth, ACP III, 5.2.2. (What tree parameter does this correspond to?)

Lecture 27

Topic du jour: Random trees (general), random walks, Brownian motion.

References: Knuth, ACP I, 2.3.4.4; G. Louchard, Brownian Motion and Algorithm Complexity, BIT v. 26, 1986, pp. 17-34; Karlin and Taylor, A First Course in Stochastic Processes (for Brownian motion);

General trees

We now consider trees in which there is no *a priori* bound on the number of children. The order of subtrees will still matter, though.

A forest is an ordered list of zero or more trees.

We can represent a forest as a binary tree. The idea is what every node "knows" its eldest child and its next youngest sibling.

[Art work goes here.]

This allows us to count the number of general trees with n nodes, since it is equal to the number of binary trees with $n-1$ nodes. This is the Catalan number

$$\frac{\binom{2n-2}{n-1}}{n}$$

Random walks

Let X_i be ± 1, with equal probability. The displacements of the sum

$$S_n = \sum_{i=1}^{n} X_i$$

form a random walk on the integers. If we plot times $n = 0, 1, 2, \ldots$ on the horizontal axis and indicate the displacments vertically, connecting the dots gives a jagged path. These paths are important combinatorial objects.

Suppose we take a tree and walk around it, starting from the root. At each node we record the distance (number of edges) to the root. Plotting these distances, we get a lattice path (random walk sample path) with the following properties:

It goes from $(0, 0)$ to $(2n-2, 0)$.

It is never negative.

Example:

If P is a random-walk path as above (never negative, returns to the origin), we can represent it as a balanced parenthesis string. The idea is to write (when the path goes up, and) when it goes down. For example, a tree in which the root has two children, and the eldest child three grandchildren, is encoded as

$$(()()())().$$

This leads to a simple recognition algorithm using a counter.

Brownian motion

The limiting form of the symmetric random walk is a stochastic process called *Brownian motion*. We are interested in it because many asymptotic results in combinatorics and algorithms are equivalent to properties of Brownian motion.

Defining properties of Brownian motion (KT p. 343):

A "sample path" of Brownian motion is a function $X(t)$, defined for all $t \geq 0$. It has $X(0) = 0$, with X continuous at 0.

If $s > 0$, then for all t, the increment $X(t+s) - X(t)$ has the normal distribution with mean 0 and variance t.

If $0 = t_0 < t_1 < \cdots < t_n$, then the increments

$$X(t_i) - X(t_{i-1})$$

are independent for $i = 1, \ldots, n$.

Theorem: Brownian motion actually exists! (Wiener, 1923.)

Additional facts

Sample paths are continuous with probability 1.

Sample paths are almost surely nowhere differentiable.

Brownian motion is a Markov process. This means that what happens after time t is independent of everything except $X(t)$.

A continuous model for random trees

Recall that we indicated how a tree could be represented as a lattice path that returns to the origin and never goes negative.

The limiting continuous process for such lattice paths is *Brownian excursion*. This is Brownian motion on $[0,1]$, contitioned to have $f(1) = 0$, and $f(t) > 0$ for $0 < t < 1$.

> Warning: This is different from the random process you get by conditioning Brownian motion to return to 0, and taking its absolute value.

The invariance principle for Brownian excursion says that if $T(0), T(1), \ldots, T(2n-2)$ are the locations of a random walk conditioned to be non-negative and return to 0,

$$\hat{T}_n(t) = \frac{T((2n-2)t)}{\sqrt{2(n-1)t}}$$

(we extend the right hand side to be piecewise linear), and f is a sample path of Brownian excursion, then

$$E[\phi(\hat{T}_n)] \to E[\phi(f)]$$

as $n \to \infty$. (In other words, conditioning doesn't affect the results.)

We can now build up a "dictionary" between trees, lattice paths, and Brownian excursion. We have

$$\text{tree height} \leftrightarrow \max T(i) \leftrightarrow \max f(t)$$

and

$$\text{internal path length} \leftrightarrow \text{area under } T \leftrightarrow \int_0^1 f(t)dt$$

This just gives the rough picture. In any real problem, scaling and/or consideration of low order terms will be required.

Lecture 28

Topic du jour: Invariance principle for Brownian motion and its connection to algorithms.

Reference: G. Louchard, Brownian Motion and Algorithm Complexity, BIT v. 26, 1986, pp. 17-34.

Symmetric random walks

Start with a particle at location 0. At each time step, take a step left or right with equal probability.

Suppose that we take n steps of this kind. The displacements give random variables

$$S_m(n) = \sum_{i=1}^{m} X_i, \qquad m = 0, \ldots, n$$

where $X_i = \pm 1$.

Brownian motion

This is the "infinitesimal" version of the random walk. Here we have a probability distribution on certain functions $X : [0, \infty) \to \mathbf{R}$, such that:

$X(0) = 0$, and X is continuous at 0.

The increment $X(t+s) - X(t)$ is normally distributed with mean 0 and variance s.

If $0 = t_0 < t_1 < \cdots < t_n$, then the increments

$$X(t_i) - X(t_{i-1})$$

are independent for $i = 1, \ldots, n$.

It is a nontrivial task to prove that such a distribution exists.

Donsker's invariance principle

Consider a random walk (start at 0, steps are equally likely to be ± 1). Scale displacement by $1/\sqrt{n}$, and time by $1/n$. The invariance principle states that as $n \to \infty$, this "looks like" Brownian motion. We give a precise formulation below.

Let $X_1, \ldots, X_n$ be i.i.d. random variables, uniform on $\{\pm 1\}$. Let

$$S_n(m) = \sum_{i=1}^{m} X_i,$$

and

$$\hat{S}_n(m/n) = \frac{S_n(m)}{\sqrt{n}}.$$

Extend $\hat{S}$ to all of $[0,1]$ by making it piecewise linear.

Let $\phi : C[0,1] \to \mathbf{R}$ be continuous, not necessarily linear. (Continuity is with respect to the max norm on $C[0,1]$. This means that if $f_n \to g$ uniformly, then $\phi(f_n) \to \phi(g)$.)

Then for every x,

$$\lim_{n\to\infty} \Pr[\phi(\hat{S}_n) \le x] = \Pr[\phi(X(t)) \le x].$$

This is called "convergence in distribution." By a standard theorem [REF?], when ϕ is also bounded on $C[0,1]$ we have

$$E[\phi(\hat{S}_n)] \to E[\phi(X(t))].$$

The result holds whenever the X_i are i.i.d. with mean 0 and variance 1, whence the term "invariance."

A simple example of invariance.

Let ϕ map f to $f(1)$. This is continuous. Since $X(1)$ is normal with mean 0 and variance 1,

$$\Pr[\sum_{i=1}^{n} X_i < xn^{1/2}] \to \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-t^2/2} dt.$$

This is the central limit theorem.

Even though ϕ is not bounded, in this case we do have (since the X_i's have mean 0)

$$\lim_{n\to\infty} E[n^{-1/2} \sum_{i=1}^{n} X_i] = 0.$$

Application of the invariance principle.

Let $f(t)$ be a sample path for Brownian motion. What do we expect the maximum value of f for $0 \le t \le 1$ to be?

Karlin and Taylor (p. 346) give the distribution of the maximum:

$$\Pr[\max_{0\le t\le 1} X(t) \le x] = \sqrt{\frac{2}{\pi}} \int_0^x e^{-u^2/2} du,$$

for $x \ge 0$.

So by differentiation the density of the maximum is

$$p(x) = \sqrt{2/\pi} e^{-x^2/2}$$

for $x \ge 0$. (This is a "half normal" distribution.) Then

$$E[X] = \int_0^\infty x p(x) dx = \sqrt{\frac{2}{\pi}}.$$

It is tempting to conclude the following. Let $X_1, X_2, \ldots$, be i.i.d. random variables, equally likely to be ± 1. For the sums $S_m = \sum_{i=1}^{m} X_i$ we have

$$E\left[\frac{\max_{m\le n} S_m}{\sqrt{n}}\right] \sim \sqrt{\frac{2}{\pi}},$$

that is,

$$E\left[\max_{m\le n} S_m\right] \sim \sqrt{\frac{2n}{\pi}}. \qquad (*)$$

This is indeed true. A combinatorial proof of it may be found in J. L. Palacios, American Statistician 62, 2008, pp. 138-140.

Application to bin packing.

The bin packing problem.

We are given X_i for $i = 1, \ldots, n$, where $0 \le X_i \le a$. Our goal is to group these numbers into sets (the "bins") summing to a or less, and minimize the number of sets.

This problem is NP-complete, so we do not expect to solve it exactly.

A heuristic algorithm for bin packing:

Call X_i "large" if it is $> a/2$, and "small" otherwise. Let

$$Y_i = \begin{cases} X_i, & \text{if } X_i \text{ is small;} \\ 1 - X_i, & \text{if } X_i \text{ is large.} \end{cases}$$

We sort the Y_i's and write under each a plus sign if Y_i came from a large item, or a minus sign if Y_i came from a small item.

We go through the + numbers (smallest first), and match each with the largest preceding − number that has not been used, should that exist. Otherwise put it in a bin by itself. At the end of the process, all unused − numbers are paired up, except possibly one.

Example: Let $a = 100$. Suppose the X_i's are

$$31, 40, 59, 26, 53, 58, 97, 93$$

the sorted Y_i's are

3	7	26	31	40	41	42	47
+	+	−	−	−	+	+	+

We put the first two into bins by themselves, and then match 41 to 40, 42 to 31, and 47 to 26. So there are five bins, which contain

97	93	40	31	26
		59	58	53

How many bins are used?

Think of the signs as describing a random walk path, where we go up 1 unit for $+$ and down 1 unit for $-$. For our example this is

[draw picture here]

The number of items that get bins to themselves is

$$\leq [\text{ max lead of } + \text{ over } -] + 1$$

(We get the extra $+1$ because there could be a leftover $-$ at the end.)

In other words, the number of bins used is

$$\frac{n + \text{ \# of ``singleton'' bins}}{2}$$

A probabilistic model for the input

We now assume that $X_1, X_1, \ldots$ are i.i.d. samples from the uniform distribution on $[0, a]$. (The analysis will actually carry through for any distribution for which the probabilities for large and small items are the same.)

Under these assumptions, the expected number of bins in the optimum packing is $\geq n/2$, since with probability $1/2$, an item will need to be in a bin by itself.

Using the invariance principle

Let OPT denote the minimum number of bins required. This is a random variable. Then

$$\begin{aligned} &\Pr[\text{ \# used } - \text{OPT} \geq \alpha\sqrt{n}] \\ \leq &\Pr[\text{ \# used } - n/2 \geq \alpha\sqrt{n}] \\ = &\Pr[\max_{m \leq n} S_m \geq \alpha\sqrt{n} + 1]. \end{aligned}$$

By the invariance principle, this converges to

$$\sqrt{\frac{2}{\pi}} \int_\alpha^\infty e^{-u^2/2} du \sim \sqrt{\frac{2}{\pi}} \frac{e^{-\alpha^2/2}}{\alpha} du$$

as $n \to \infty$.

Therefore, with very high probability, the relative error incurred by using the fast heuristic is $O(n^{-1/2})$.

Using (*), we can say more. We have

$$E[\text{ \# of bins used }] = n/2 + O(\sqrt{n}).$$

One conclusion we can draw from this is that

$$E[\text{OPT}] \sim \frac{n}{2},$$

since the optimum is surely less than or equal to what our heuristic constructs.

As a measure of performance, we have

$$\frac{E[\text{ \# used }]}{E[\text{ optimum }]} = 1 + O(n^{-1/2}).$$

I first saw this in a lecture by R. M. Karp. Possible citation: G. S. Lueker, An average-case analysis of bin packing with uniformly distributed item sizes, Technical Report 181, UC Irvine, 1982.

Computation of mean IPL for a random tree

We note that if T is a general tree, then the corresponding lattice path (after piecewise linear extension to a continuous function) has area

$$A = 2I(T) - (n-1).$$

Let $f(t)$, for $0 \le t \le 1$, be a Brownian excursion. The distribution of $Y = f(t)$ is known (Ito/McKean p. 76). It has the density

$$p(y) = \frac{2y^2 \exp(\frac{-y^2}{2t(1-t)})}{\sqrt{2\pi t^3 (1-t)^3}}.$$

This is a *Maxwell* distribution, which gives the length of a vector in 3-space, whose components are i.i.d. random samples from the normal distribution with mean 0 and variance $t(1-t)$.

Writing $\sigma^2 = t(1-t)$, we compute its mean value as

$$E[f(t)] = \int_0^\infty \frac{2y^3 e^{\frac{-y^2}{2\sigma^2}}}{\sqrt{2\pi\sigma^3}} dy = \int_0^\infty \frac{4\sigma^2 u e^{-u} du}{\sqrt{2\pi\sigma^3}}$$

$$= \sqrt{\frac{8}{\pi}}\sigma^{1/2} = \sqrt{\frac{8}{\pi}}t^{1/2}(1-t)^{1/2}.$$

The area under $f(t)$ is therefore

$$E[\int_0^1 f(t)dt] = \int_0^1 E[f(t)]dt = \frac{2\sqrt{2}}{\sqrt{\pi}}\int_0^1 t^{1/2}(1-t)^{1/2}dt.$$

The integral is a standard one called the *Euler beta integral*; indeed, we have

$$\int_0^1 t^{a-1}(1-t)^{b-1}dt = \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)}.$$

Using $\Gamma(3/2) = \sqrt{\pi}/2$, we get for the area

$$E[\int_0^1 f] = \sqrt{\frac{\pi}{8}}.$$

Now we can work our way back to trees. For the lattice path we have

$$E[\text{ area }] = (2n)^{3/2}\sqrt{\frac{\pi}{8}} \sim 2E[I(T)].$$

Therefore, for a random tree, we guess that

$$E[I(T)] \sim \frac{\sqrt{\pi}}{2}n^{3/2}.$$

This can be proved combinatorially. See D. Merlini et al., The area determined by under-diagonal lattice paths, Proc. CAAP, Lecture Notes in Computer Science, v. 1059, 1996, pp. 59-71. See also R. Chapman, Moments of Dyck Paths, Disc. Math. 204, 1999, pp. 113-117.

The expected tree height can be computed in the same way. For this you need the distribution of the maximum of a Brownian excursion (Durrett and Inglehart, Ann. Prob. 1977), which has expected value $\sqrt{\pi/2}$. The result is

$$E[\text{height}(T)] \sim \sqrt{2n}\sqrt{\frac{\pi}{2}} = \sqrt{\pi}n^{1/2}.$$

Notes.

The invariance principle is called the functional central limit theorem by some authors. The version we quoted is a simplified version of a theorem appearing on p. 77 of Freedman, Brownian Motion and Diffusion. Billingsley, Probability and Measure, has what looks like a "convergence in probability" version on p. 548.

There is a connection between 2-ordered permutations (important in sorting) and Brownian bridge.

CS 709
Eric Bach
Spring 2001

Lecture 29

Topic du jour: Random graph models

Reference: B. Bollobás, Random Graphs. R. M. Karp, lecture notes, 1982.

Overview

We want to study the "expected" or "typical" inputs for graph algorithms. It's not obvious exactly how to do this, so there are several competing models.

Uniform: Choose uniformly from among the isomorphism classes of n-vertex graphs. This is hard to study (in part because the isomorphism classes are themselves complicated), and doesn't seem to model any natural process. We will say no more about it.

$G_{n,p}$: Start with n vertices. For each possible edge, include it independently with probability p.

d-out: For each vertex, choose d neighbors by sampling without replacement. (This is a natural model for directed graphs of fixed degree.)

Evolution: Imagine an urn full of edges. Sample from the urn and add them to the graph one by one. If we stop after a fixed number of samples, this gives us a "sampling without replacement" version of $G_{n,p}$.

Intuition: Graph properties depend on edge density

The theory of random graphs was popularized in 1960 by Erdős and Rényi, with a paper entitled "On the Evolution of Random Graphs." Their most surprising finding was that the time at which certain properties appear is sharply predictable. You can think of this as analogous to phase transitions in the theory of complex physical systems.

The following table shows the point at which various properties appear with high probability.

m (number of edges)	$p = m/\binom{n}{2}$	Property	Page
$o(n)$	$o(1/n)$	All components trees	36
$o(n^{\frac{k-2}{k-1}})$	$o(n^{-k/(k-1)})$	No k-vertex trees	23
cn, $c < 1/2$	c/n, $c < 1$	All components trees or 1-trees	44
$n/2$	$1/n$	Largest component $O(n^{2/3})$	49
cn, $c > 1/2$	c/n, $c > 1$	Largest component $O_c(n)$	56
$n \log n/2$	$(\log n)/n$, $c > 1$	Giant component + isolated vertices	57
$n(\log n + c)/2$, $c > 0$	$(\log n + c)/n$,	Connected	57
$\omega(n \log n)$	$\omega(\frac{\log n}{n})$,	Degrees almost equal	58

Here we write $\omega(f)$ for any function that goes to infinity faster than f.

Page numbers are from Erdős and Rényi.

Many of these results hold for the $G_{n,p}$ model, so for comparison, we have given the edge density p.

Isolated vertices for $G_{n,p}$ (warmup)

What's the expected number of isolated vertices?

Let p denote the probability of "edgeness," with $q = 1 - p$.

Set $X_i = 1$ if the i-th vertex is isolated, 0 otherwise. Note that

$$E[X_i] = \Pr[X_i = 1] = q^{n-1}.$$

Therefore

$$E[\text{ \# of isolated vertices }] = nq^{n-1} = n(1-p)^{n-1} \leq ne^{-(n-1)p}.$$

This suggests that we should expect to see isolated vertices when p is small compared to $\log n/n$, but not when p exceeds $\log n/n$.

Isolated vertices for $G_{n,p}$ (threshold result).

We will prove that if $p = \frac{\log n + c}{n}$ and $n \to \infty$, then the probability that no vertex is isolated is asymptotic to

$$e^{-e^{-c}}.$$

This function is monotonically increasing, from 0 (when $c = -\infty$) to 1 (when $c = +\infty$). Therefore, this is the distribution function of a random variable. This is called the *Gumbel extreme-value distribution* and plays a role in order statistics.

Its mean value is

$$\int_{-\infty}^{\infty} x e^{-x} e^{-e^{-x}} dx = \gamma.$$

Most of the mass is concentrated in a narrow band around $c = 0$. For example, 99.9% of the mass lies in $[-2, 7]$.

Inclusion-exclusion inequalities

Let E_i be the event that the i-th vertex is isolated, and put

$$P_n = \Pr[\bigcup_{i=1}^{n} E_i].$$

(That is, P_n is the chance that some vertex is isolated.)

We have

$$\begin{aligned} P_n &\leq \sum_i \Pr[E_i] \\ P_n &\geq \sum_i \Pr[E_i] - \sum_{i<j} \Pr[E_i \cap E_j] \\ P_n &\leq \sum_i \Pr[E_i] - \sum_{i<j} \Pr[E_i \cap E_j] + \sum_{i<j<k} \Pr[E_i \cap E_j \cap E_k], \end{aligned}$$

and so on. Let's put

$$S_r(n) = (-)^{r+1} \sum_{1 \leq i_1 < i_2 < \cdots < i_r \leq n} \Pr[E_{i_1} \cap \ldots \cap E_{i_r}].$$

Then for $k \geq 1$,

$$\sum_{r=1}^{2k} S_r(n) \leq P_n \leq \sum_{r=1}^{2k-1} S_r(n)$$

Limits

The probability that any particular r vertices are all isolated is

$$q^{(n-1)r - \binom{r}{2}} = q^{nr} q^{-r-\binom{r}{2}},$$

since there are $n-1$ vertices connected to each of these r vertices, thus counting $\binom{r}{2}$ edges twice. So

$$S_r(n) = \pm \binom{n}{r} q^{nr} q^{-r-\binom{r}{2}}.$$

For fixed r and $n \to \infty$, we have

$$\binom{n}{r} = \frac{n^{\underline{r}}}{r!} \sim \frac{n^r}{r!},$$

and (since $p = (\log n + c)/n$),

$$q^{nr} = (1 - \frac{\log n + c}{n})^{nr} \sim \exp(-(\log n + c)r) = \frac{e^{-cr}}{n^r}.$$

Since $q \to 1$, any fixed power of q will go to 1 as well, and we therefore have

$$S_r(n) \sim (-)^{r+1} \frac{e^{-cr}}{r!}.$$

This gives

$$\sum_{r=1}^{2k} (-)^{r+1} \frac{e^{-cr}}{r!} \leq \liminf P_n \leq \limsup P_n \leq \sum_{r=1}^{2k-1} (-)^{r+1} \frac{e^{-cr}}{r!}$$

Since this holds for all k, we are now entitled to let $k \to \infty$ and conclude

$$\lim P_n = \sum_{r=1}^{\infty} (-)^{r+1} \frac{e^{-cr}}{r!} = 1 - \exp(-e^{-c}).$$

Notes

It can be shown that the same threshold result holds for connectedness. The idea is to prove that with probability $\to 1$, the graph is connected iff it has no isolated vertices.

For moments of the Gumbel distribution, see Kendall and Stuart, v. 1, Exercise 14.4.

One difficulty with $G_{n,p}$ and similar models is that they do not exhibit the same "clustering" as certain real-world graphs seem to. For an introduction to models that attempt to remedy this, see B. Hayes, American Scientist, March-April 2000, pp. 104-109,

Two nice surveys of random graphs: G. Grimmett, in Selected Topics in Graph Theory 2, ed. Beineke and Wilson; Karoński, J. Graph Theory v. 6, 1982, pp. 349-389.

Erdős and Rényi's paper is in Pub. Math. Inst. Hungarian Acad. Sci., v. 5, 1960, pp. 17-61.

There should be a good reference for the behavior of $\lim(1 - z/n)^n$ when z varies with n. In the mean time, note that for $-2/3 < z < 0$ we have

$$-z(1 + z) \leq \log(1 - z) \leq -z.$$

Multiply this by n and put $z = (\log n + c)/n$. Upon exponentiation, this becomes (after simplification)

$$\frac{e^{-c}}{n} \left(1 + O\left(\frac{\log n}{n}\right)\right) \leq \left(1 - \frac{\log n + c}{n}\right)^n \leq \frac{e^{-c}}{n}.$$

Therefore, for fixed r and $n \to \infty$,

$$\left(1 - \frac{\log n + c}{n}\right)^{nr} \sim \frac{e^{-cr}}{n^r}$$

as claimed.

Lecture 30

Topic du jour: Minimum spanning trees

Definitions

Let G be a complete graph on n vertices.

Then G has $\binom{n}{2}$ edges.

A subgraph T of G is a *spanning tree* if T is connected and has $n-1$ edges.

We give each edge a nonnegative *cost*. If T is a spanning tree, then

$$\text{cost}(T) = \sum_{e \in T} \text{cost}(e).$$

The problem is to construct a tree T that minimizes the cost.

Greedy algorithm

One standard algorithm for this problem is the *greedy algorithm*:

Sort the edges by cost, so that

$$\text{cost}(e_1) \leq \text{cost}(e_2) \leq \cdots.$$

Begin with an empty forest F.

For $i = 1, 2, \ldots$: If $F + e_i$ forms a cycle, continue with the next edge. Otherwise, add e_i to F, and halt the algorithm if F has $n-1$ edges.

Example:

Some facts about the algorithm (see any book on algorithms):

It actually constructs a minimum spanning tree.

Using heaps, we can avoid sorting the edges.

With dynamic equivalence relations (union-find structures) it is easy to decide if $F + e_i$ forms a cycle.

We'll focus on a different question: how many edges are considered? In other words, how many times will we go around the loop?

There are graphs for which the algorithm will look at $\sim n^2/2$ edges. An example is a complete graph on $n-1$ points, with all edges of cost 1, each of whose vertices is joined to an new vertex by an edge of cost 2.

We will prove that on average, $O(n \log n)$ edges are examined. This is suggested by Erdős-Rényi theory, since a random graph becomes connected, with high probability, after this many edges appear.

Expected behavior of greedy algorithm

Assume the edge costs are drawn independently from some distribution. We will get the same results for any continuous distribution, so we may as well assume that the edge costs are i.i.d. from the uniform distribution on $[0, 1]$.

This puts an ordering on the edges, and our question is equivalent to the following one: suppose we have an urn full of edges. Draw from the urn (without replacement) and use these edges to build a tree, throwing away any that make a cycle with the partial tree already built. How many edges are needed?

We make 2 simplifications to ease the analysis:

Sample with replacement. This may make the process run longer, but it will not change the ultimate result. (Any "duplicate" sample is already in the tree, or forms a cycle with some part of the tree. So it will be rejected.)

Focus on the growth of *one* tree T. Equivalently, if an edge doesn't connect to the tree we are trying to grow, put it back in the urn. (This also can only increase the running time.)

Expected waiting time to enlarge the tree

Suppose T has m vertices, with $1 \le m \le n-1$. Then there will be $m' = n - m$ vertices outside T. The chance of drawing an edge that links T to $\bar{T}$ is

$$\Pr[\text{ extend } T \text{ }] = \frac{mm'}{\binom{n}{2}}.$$

The waiting time for a success is geometric, with mean value equal to the inverse of the probability of success. Therefore, the expected time to extend the tree is

$$\frac{\binom{n}{2}}{mm'}.$$

Total expected waiting time

The expected time to grow the tree to n vertices is

$$\begin{aligned} 1 + \sum_{m=1}^{n-1} \frac{\binom{n}{2}}{m(n-m)} &= 1 + \frac{n-1}{2} \sum_{m=1}^{n-1} \frac{n}{m(n-m)} \\ &= 1 + \frac{n-1}{2} \sum_{m=1}^{n-1} \left(\frac{1}{m} + \frac{1}{n-m} \right) \\ &= (n-1)H_{n-1} + 1 \sim n \ln n. \end{aligned}$$

Evidently, this is an upper bound on the number of edges considered by the minimum spanning tree algorithm.

We can get an exact result if we appeal to the work of Erdős and Rényi. They showed that if we choose

$$m = \frac{n(\log n + c + o(1))}{2}$$

edges at random, then the probability that the graph is connected is asymptotic to

$$\exp(-e^{-c})$$

as $n \to \infty$. Therefore, with high probability, the algorithm uses $\sim n \log n$ edges and no more.

Estimates for the cost of the MST

The cost of the MST does depend on the distribution of costs. For simplicity we shall assume that the costs are i.i.d. uniform on [0,1].

Lower bound:

We use the theory of order statistics.

Suppose that $X_1, X_2, \ldots, X_m$ are i.i.d. samples from the uniform distribution on $[0, 1]$. The sorted samples

$$X_{(1)} \leq X_{(2)} \leq \cdots \leq X_{(m)}$$

are called the *order statistics*. They are distinct, with probability 1.

Theorem:

$$E[X_{(i)}] = \frac{i}{m+1}.$$

To prove this, choose $m+1$ random points on a circle of diameter 1. Cut the circle at one of the points. The remaining points are i.i.d. random samples, uniformly distributed on $[0, 1]$. From this it is easy to see that all the "gaps" $X_{(i)} - X_{(i-1)}$ have the same distribution, so their common mean value is $1/(m+1)$.

From this we get

$$E[\text{ cost of MST }] \geq E[\text{ cost of cheapest } n-1 \text{ edges }],$$

and this is

$$\sum_{i=1}^{n-1} \frac{i}{\binom{n}{2}+1} = \frac{\binom{n}{2}}{\binom{n}{2}+1} \sim 1.$$

Upper bound:

We can get a quick and dirty $O(\log n)$ upper bound as follows. Generate a minimum spanning tree for $n-1$ vertices, then connect this to the n-th vertex by the minimum possible edge. This gives some spanning tree, not necessarily the minimum. If E_n is the expected value of the minimum spanning tree on n vertices, we have (by taking expected values)

$$E_n \leq E_{n-1} + \frac{1}{n}; \qquad E_1 = 0.$$

This gives $E_n \leq H_n - 1 \sim \log n$.

With a little more work, we can obtain $O(1)$.

Lemma: If $m_1, \ldots, m_r$ are positive integers with $\sum m_i = m$, then

$$\sum_{i=1}^{r} \binom{m_i}{2} \leq \binom{m}{2}.$$

To prove this, count the edges in a graph with m [sic] vertices which are divided into subsets of sizes $m_1, \ldots, m_r$.

We will now modify the MST algorithm so that it chooses the tree first and then chooses costs for the edges. Start with an urn full of edges, and an empty forest F. Sample from the urn without replacement, assigning each edge a serial number called its *rank*. Any edge that does not form a cycle with the edges already in F is added to the forest. Quit drawing from the urn when F contains $n-1$ edges. Then, generate $U[0,1]$ order statistics

$$X_{(1)} < X_{(2)} < \cdots < X_{(\binom{n}{2})},$$

and assign any tree edge with rank r the cost $X_{(r)}$. Because the greedy algorithm only requires the ranks and not the actual costs, the end result is the same.

Summing over i and using the expected values for order statistics, we get

$$E[\text{cost of MST}] = E\left[E[\text{cost of MST}|r_1, \ldots, r_{n-1}]\right] = E\left[\sum_{i=1}^{n-1} \frac{r_i}{\binom{n}{2}+1}\right],$$

Here, r_i denotes the rank of the i-th edge added to the forest.

So we must find the expected ranks of the edges added to the tree. We analyze this by dividing the process into stages. The i-th stage starts when the forest has $i-1$ edges, and stops when a new edge is added.

How many edges are sampled in the i-th stage? We can assume sampling with replacement, since this will only increase the running time, without changing the rank of the edge added. (Repeated edges have lower rank.) By the lemma, there are at least

$$\binom{n}{2} - \binom{i-1}{2},$$

edges that can extend the forest. So we expect to use at most

$$\frac{\binom{n}{2}}{\binom{n}{2} - \binom{i-1}{2}}$$

edges in the i-th stage.

For $i = 1, \ldots, n-1$, let r_i be the rank of the i-th edge added to the forest. By the above reasoning, we have

$$E[r_i] \leq E[r_{i-1}] + \frac{\binom{n}{2}}{\binom{n}{2} - \binom{i-1}{2}}, \qquad E[r_1] = 1.$$

So by induction,

$$E[r_i] \leq \sum_{j=0}^{i-1} \frac{\binom{n}{2}}{\binom{n}{2} - \binom{j}{2}}.$$

Using this gives

$$E[\text{cost of MST}] \leq \frac{2\binom{n}{2}}{\binom{n}{2}+1} \sum_{i=1}^{n-1} \sum_{j=0}^{i-1} \frac{1}{n(n-1) - j(j-1)}.$$

To obtain a closed form for this sum, first note that $n(n-1) - j(j-1) = (n-j)(n+j-1)$. Then interchange the order of summation and find

$$\sum_{i=1}^{n-1} \sum_{j=0}^{i-1} \frac{1}{n(n-1) - j(j-1)} = \sum_{j=0}^{n-2} \frac{n-j-1}{(n-j)(n+j-1)}$$

$$= \sum_{j=0}^{n-2} \frac{1}{n+j-1} - \frac{1}{2n-1} \sum_{j=0}^{n-2} \left(\frac{1}{n-j} + \frac{1}{n+j-1} \right)$$

$$= H_{2n-3} - H_{n-2} - \frac{1}{2n-1} \left(H_{2n-3} - 1 + \frac{1}{n-1} + \frac{1}{n} \right).$$

Bottom line: The MST has expected cost

$$\leq \frac{2\binom{n}{2}}{\binom{n}{2}+1}\left(H_{2n-3}-H_{n-2}-\frac{1}{2n-1}\left(H_{2n-3}-1+\frac{1}{n-1}+\frac{1}{n}\right)\right),$$

so

$$E[\text{cost of MST}] \leq 2\log 2 + O((\log n)/n).$$

What's the correct answer?

We have proved that the MST has expected cost $\Theta(1)$. More precisely,

$$1 + O(1/n) \leq E[\text{ cost of MST }] \leq 2\log 2 + O(\frac{\log n}{n}).$$

Which is correct? Frieze (Disc. Appl. Math. 1985) proved that

$$E[\text{ cost of MST }] \sim \sum_{n\geq 1}\frac{1}{n^3} = \zeta(3) = 1.202057...$$

Lecture 31

Topic du jour: Matching in graphs

Feller, v. 1 (for coupon collection); Lovasz and Plummer, Matching Theory (for everything you wanted to know about matching); Angluin and Valiant, JCSS 1979 (for the nobleman's algorithm).

The coupon collection problem

This is a "must know" result in probability theory, with many applications to algorithms.

Suppose we throw balls into n bins, uniformly and independently. We seek information about the time T by which each bin has at least one ball.

Expected value:

If i bins are occupied, the probability of a ball's landing in a "new" bin is $(n-i)/n$. Hence the waiting time for extending the number of occupied bins is geometrically distributed, with expected value $n/(n-i)$.

Therefore,

$$E[T] = \sum_{i=0}^{n} \frac{n}{n-i} = nH_n \sim n \log n.$$

Variance:

If we note that the individual waiting times are independent, we can prove similarly that

$$\sigma^2(T) = n^2 \sum_{i=1}^{n} \frac{1}{i^2} - n \sum_{i=1}^{n} \frac{1}{i}.$$

The first sum, if taken over all positive integers, is the value at $s = 2$ of the *Riemann zeta function*

$$\zeta(s) := \sum_{i \geq 1} \frac{1}{i^s}.$$

Euler showed that $\zeta(2) = \frac{\pi^2}{6}$. So

$$\sigma^2(T) = n^2 \frac{\pi^2}{6} + O(n \log n).$$

Tail bound:

Let $c > 0$. We have

$$\Pr[T > t] = \Pr[\text{ some bin not occupied after } t \text{ trials }] \leq n(1 - 1/n)^t,$$

and for $t = cn \log n$ this bound is $\leq 1/n^{c-1}$.

Limiting distribution:

Although T is a sum of independent random variables, its limiting distribution is *not* normal. We state the following result, due to Erdős and Rényi, without proof. If c is a constant, then

$$\lim_{n\to\infty} \Pr[T \geq n(\log n + c)] = 1 - e^{-e^{-c}}.$$

Matching in graphs

Let $G = (V, E)$ be an undirected graph. A *matching* is a set of edges that don't share any vertices.

Example: In the graph

$$\begin{array}{ccc} A & - & B \\ | & & | \\ C & - & D \end{array}$$

the edges $\{A, B\}$ and $\{C, D\}$ form a matching.

Some people (Harary in particular) call these *1-factors*.

A *perfect matching* is one with $|V|/2$ edges, the maximum possible number.

Many applications deal with bipartite graphs, in which the edges can be thought of as a subset of $X \times Y$.

The traditional story is that X is a set of men, Y a set of women, with the edge $e = (x, y)$ indicating that x and y are compatible, say as partners for a big dance. The goal is to pair off as many people as possible.

These are often called *assignment problems*, from the problem of assigning workers to jobs.

Algorithmic problems:

Find the largest possible matching.

Find the optimal (least-cost) matching, in the model where each edge has a cost.

These can be solved in polynomial time, but not particularly quickly. For example, one standard algorithm for bipartite matching (maximum flow, with breadth-first search for augmenting paths) uses $O(n^3)$ steps when $|X| = |Y| = n$. For this reason, fast heuristics are of interest.

The nobleman's algorithm

This is a heuristic to construct a (large, we hope) matching in a bipartite graph.

The algorithm works by a series of *proposals*. The idea is to simulate the notorious *droit de seingeur*, familiar from The Marriage of Figaro. (We use a sanitized version, since this is not a tacky opera plot but a university course.) During the algorithm,

women will become "engaged" to various men. (One at at time of course!) If a man proposes to a woman, she immediately accepts the proposal and returns her former inamorato to the pool of unattached men. This man then becomes the next to propose.

We imagine that each man has a list of compatible women, and crosses each off as she rejects him. Clearly, one of two things must happen: The algorithm stops (because some man has no one else to propose to), or a perfect matching is formed.

A formal description:

$X :=$ first man
while there are unengaged couples:
    if X's list is empty the algorithm stops
    else
        $y :=$ first woman on X's list
        if y is unattached:
            pair X and y
            $X :=$ next available man (if any)
        else (y is engaged to Z)
            dissolve the pair $Z - y$ and cross y off Z's list
            engage X to y
            $X := Z$

This is very similar to the Gale-Shapley algorithm for stable marriages, but different in two ways. First, there are no preferences. Second, every proposal is accepted.

Note that the result will depend on the way in which men are ordered, as well as the orderings within the men's lists.

Example: Let G be

$$\begin{array}{ccc} A & - & a \\ & \backslash & \\ B & - & b \end{array}$$

If this is presented as

$$\begin{aligned} A &: \ a\ b \\ B &: \ b \end{aligned}$$

A is the first suitor. He proposes to a, and this is accepted. Then B becomes the suitor, and his proposal to b is accepted. So we have found a perfect matching.

G could also be presented as

$$\begin{aligned} B &: \ b \\ A &: \ b\ a \end{aligned}$$

Then, B is the first suitor, whose proposal to b is accepted. After that A becomes the suitor, who displaces B. At this point, B must propose again, but he is out of choices. So the algorithm stops with the (suboptimal) matching $A - b$.

Facts about the algorithm.

Once a woman becomes engaged, she stays engaged.

The algorithm finds a perfect matching iff every woman receives a proposal.

The number of proposals is $\leq |E|$ (the number of edges), so this algorithm runs in linear time.

The nobleman's algorithm for a complete bipartite graph.

For the complete bipartite graph $K_{n,n}$, there is an obvious "greedy" strategy: Make each new man propose to an unattached woman. So we'd like to know how close the nobleman's algorithm can come to this ideal.

We assume that the list of each man is randomly ordered. (You can think of this as an extra randomizing step in the algorithm.)

To make the algorithm easier to analyze, we introduce "male amnesia." This means that each man, when he is the proposer, chooses one of the n women uniformly. We include the additional rule that a duplicate proposal is rejected. By the principle of deferred decisions, this is equivalent, as far as the sequence of accepted proposals is concerned, to running the nobleman's algorithm with lists that are randomly ordered.

The matching gets larger by one edge each time a woman receives her first proposal. Thinking of these proposals as balls thrown into n bins, the number of proposals this algorithm makes is identical to the number of "throws" for the coupon collection problem.

So, if the nobleman's algorithm is run on $K_{n.n}$, with the men's lists randomly ordered, we can state the following. First, the expected number of proposals used to find a perfect matching is at most

$$E[\text{ waiting time to collect } n \text{ coupons }] = nH_n.$$

Furthermore, if $t = Cn \log n$, the probability that more than t proposals are made is upper bounded by

$$\Pr[\text{ coupon collection needs} > t \text{ balls }] \leq \frac{1}{n^{C-1}}.$$

Notes.

Nice references on probabilistic aspects of Gale-Shapley matching: D.E. Knuth, Mariages Stables et Leurs Relations avec d'Autres Problèmes Combinatoires, Presses Univ. Montreal, 1976. [AMS has published an English translation, with updates by the author.]

Angluin and Valiant's matching heuristic was originally for undirected graphs. The bipartite version presented here and in the next lecture is from a course by R. M. Karp (CS 292, 1982).

The limiting distribution for coupon collection is in Erdős and Rényi, On a classical problem of probability theory, Pub. Math. Inst. Hungarian Acad. Sci., v. 6, 1961, pp. 215-220. See also Section 5.4.1 of Mitzenmacher and Upfal, Probability and Computing (2005).

The various large deviation bounds (Chernoff, Azuma-Hoeffding, BRW bounds, etc.) should be treated in one place. Basically they are all using the same idea.

Lecture 32

Topic du jour: Stochastic ordering, large deviation bounds, the nobleman's algorithm on random bipartite graphs.

References: H. Chernoff, Ann. Math. Stat. 1952; Angluin and Valiant, loc. cit.

Stochastic ordering

It is useful to have a notion of when one random variable "tends to be larger" than another. Pointwise comparision is too restrictive, in part because this requires a common sample space.

Defn: For two random variables, $X \leq_s Y$ means that $\Pr[X \leq z] \geq \Pr[Y \leq z]$ for all z.

Note that the sense of the inequality is reversed when we go to the distribution functions.

You can remember stochastic ordering this way. It means that one random variable is more likely to be small than another.

Example: $X \sim B(m,p)$ (binomial random variable) and $Y \sim B(n,p)$, with $m \leq n$. (In this case, the inequality holds pointwise, since we can sample X and then flip $n-m$ additional coins to get Y.)

Large deviation estimates

Suppose that Y is a sum of n i.i.d. random variables with finite second moments. The central limit theorem tells us that Y will tend to be $\Theta(\sqrt{n})$ away from its mean, which is proportional to n. What about the probability of observations that are further out, say $\Theta(n)$ away from the mean?

The result we will prove should be considered any time you have a sum of independent random variables, and can estimate their moment generating functions. It is often called the Chernoff bound, although the story involves more authors as well.

To illustrate, we consider $Y = \sum_{i=1}^{n} X_i$, where X_i are i.i.d. Bernoulli trials with expected value p. Put $q = 1-p$ as usual, and $\lambda = E[Y] = np$.

Our goal is to bound the probability that $Y \geq k$. We use a trick that is exceedingly valuable for estimates: get an inequality that has a parameter in it, and optimize.

Let

$$H(y) = \begin{cases} 1, & \text{if } y \geq k; \\ 0, & \text{if } y < k. \end{cases}$$

Note that $H(y) \leq e^{\alpha(y-k)}$, for any $\alpha \geq 0$.

Then

$$\Pr[Y \geq k] = E[H(Y)] \leq E[e^{\alpha(\sum X_i - k)}] = e^{-\alpha k}(E[e^{\alpha X}])^n,$$

where we have used the law $E[Z_1 \cdots Z_n] = E[Z_1] \cdots E[Z_n]$, valid for independent random variables Z_i. (Here, X denotes a random variable with the same distribution as X_i.)

We now minimize the right hand side with respect to α. This is a straightforward calculus problem (it helps to take logs first). The result can be put in the form

$$e^\alpha = \frac{kq}{(n-k)p}.$$

(If you just want to verify the bound, you can ignore the optimization and accept that this is a "good" value of α.)

Note that if $k \geq np$ $(= \lambda)$, we have $k(p+q) \leq np$, which is equivalent to

$$\frac{kq}{(n-k)p} \geq 1.$$

This verifies that $\alpha \geq 0$, as required.

The moment generating function of X is

$$E[e^{\alpha x}] = pe^\alpha + q = \frac{nq}{n-k}.$$

Using our bound,

$$\begin{aligned}
\Pr[\sum X_i \geq k] \leq & \left(\frac{(n-k)p}{kq}\right)^k \left(\frac{nq}{n-k}\right)^n \\
= & \left(\frac{np}{k}\right)^k \left(\frac{nq}{n-k}\right)^{n-k} \qquad (*) \\
= & \left(\frac{\lambda}{k}\right)^k \left(\frac{(n-k)-(\lambda-k)}{n-k}\right)^{n-k} \\
\leq & \left(\frac{\lambda}{k}\right)^k e^{-\lambda+k} \\
= & e^{-\lambda}\left(\frac{\lambda e}{k}\right)^k
\end{aligned}$$

Note that this decays exponentially in k, as soon as k exceeds (say) 3λ.

There is a corresponding bound for the left tail. Observe that

$$\sum X_i \leq k \Leftrightarrow \sum \bar{X}_i \geq n-k,$$

where $\bar{X}_i = 1 - X_i$ are i.i.d. Bernoulli trials with expected value q. The intermediate result (*) is symmetric under the transformation $(p,k) \leftrightarrow (q, n-k)$. Therefore, if $k \leq \lambda$,

$$\Pr[\sum X_i \leq k] \leq e^{-\lambda}\left(\frac{\lambda e}{k}\right)^k.$$

Review of nobleman's algorithm

Start with a bipartite graph G. We think of V as composed of two disjoint sets X (men) and Y (women). An edge indicates that man x and woman y are compatible.

Last time we discussed a heuristic for finding a matching (not necessarily perfect). The algorithm constructs a matching via a sequence of "proposals." For details see the last lecture.

We showed last time that for an $n \times n$ complete bipartite graph, the expected number of proposals is $O(n \log n)$.

Our goal is now to show that a "random" bipartite graph with $O(\log n)$ edges per node will have a perfect matching, with high probability.

We will use a variant of the d-out model, in which each x in X chooses (with replacement) d random neighbors.

Technically, we do not have a graph but a multigraph in which the neighbors of vertices in X are ordered. Because of this we need to be clear about what happens with repeated edges. We adopt the convention that such duplicate proposals are always accepted.

Failure probability analysis (d-out with replacement model). I think this is due to R. M. Karp.

Let $n = |X|$, and choose

$$d = 5 \log n.$$

We will pretend that this and similar values are integers. We prove that if the algorithm fails to find a perfect matching with probability $O(n^{-1/2})$.

Consider the events

$$A := \{ \text{ more than } 2n \log n \text{ proposals } \}$$

and

$$B_i := \{ \text{ the } i\text{-th man makes } > 5 \log n \text{ proposals } \}.$$

The d-out with replacment model exactly simulates male amnesia, since we can think of the next entry for each man's list being generated as it is needed.

So, by the coupon collection tail bound in the last lecture (take $C = 2$),

$$\Pr[A] \leq \frac{1}{n}.$$

Furthermore, the algorithm fails iff the event B_i occurs for some i. So by the iterated expectation formula,

$$\Pr[\text{ failure }] \leq \Pr[\bigcup_{i=1}^{n} B_i | \bar{A}] + \Pr[A].$$

Now, consider the structure of the proposals. The i-th man becomes the proposer if he enters the game, and with probability $\leq 1/n$ at each trial thereafter, since any other man will choose his fiancée with probability exactly $1/n$. Each time he becomes the proposer, he attempts to make one proposal.

Let X_i be the number of proposals made by man i. Under conditioning by $\bar{A}$, we have

$$X_i \leq_s 1 + Y_i,$$

where Y_i is a binomial$(2n \log n, 1/n)$ random variable. (Even if the algorithm stops, we can imagine proposals being made until the clock hits $2n \log n$.)

The Chernoff upper tail bound states that if Y is a binomial(N, p) random variable, we have

$$\Pr[Y \geq k] \leq e^{-\lambda} \left(\frac{\lambda e}{k} \right)^k,$$

when $k \geq \lambda := Np$.

Use this result with $p = 1/n$, $\lambda = 2 \log n$, and $k = 5 \log n$. Then,

$$\begin{aligned} \Pr[\ i\text{-th man proposes} \geq 5 \log n + 1 \text{ times } |\bar{A}] &\leq \Pr[Y_i \geq 5 \log n] \\ &\leq e^{-\lambda} \left(\frac{\lambda e}{k} \right)^k \\ &= e^{-2 \log n} \left(\frac{2e}{5} \right)^{5 \log n} \\ &= n^{-2} n^{5 \log(2e/5)} \\ &< n^{-3/2}. \end{aligned}$$

(We note that $5 \log(2e/5) = 0.418546... < 1/2$.)

By the union bound,

$$\Pr[\bigcup_{i=1}^{n} B_i | \bar{A}] \leq n \Pr[B_i | \bar{A}] \leq n^{-1/2}.$$

This gives an overall failure probability bound of

$$\frac{1}{\sqrt{n}} + \frac{1}{n} \sim \frac{1}{\sqrt{n}}.$$

Notes.

The earliest source I know of for stochastic ordering is the book of E. L. Lehmann, Testing Statistical Hypotheses (1959).

Before Chernoff, Bernstein (1920s) and Cramer (1938) had also used the MGF of a random variable to estimate tail probabilities for i.i.d. sums. Chernoff's actual estimate (for Bernoulii trials) was the inequality (*) above. Generalizations to martingales appear in Hoeffding [REF] and Azuma [REF].

Erdős and Rényi [Pub. Math. Inst. Hungarian Acad. Sci., 1964] gave the following result for $B_{n,p}$ (start with $|X| = |Y| = n$, and throw in each of the n^2 possible edges with probability p). If $p = (\log n + c + o(1))/n$, then

$$\Pr[\ B_{n,p} \text{ has a perfect matching }] \sim \exp(-2e^{-c}).$$

For the nobleman's algorithm, one can prove the same failure probability bound for the $B_{n,p}$ model (bipartite analog of $G_{n,p}$) when $p = 10(\log n)/n$. The proof is the same, with the addition of an additional failure mode (not enough edges).

In the failure probability calculation, the (false) integrality assumption can be replaced by appropriate rounding. (Explain.)

Lecture 33

Topic du jour: Shortest paths.

References: P. M. Spira, SIAM J. Computing 1973. We follow a lecture by R. M. Karp (CS 292, 1982).

The shortest path problem

Another algorithm from the "canon" is the problem of finding shortest paths in a graph.

Let (d_{ij}) be a matrix of non-negative numbers. We interpret d_{ij} to be the distance between vertex i to vertex j.

Let s be a designated source vertex. Our goal is to find a shortest path from s to v, for all other vertices v. The result is most conveniently presented in the form of a tree. The tree has root s, and if $v \to w$ is an edge (from parent to child), we interpret this to say that the shortest path from s to w goes along tree edges to v, then goes directly from v to w.

Dijkstra's algorithm

Let the vertices be $\{1, \ldots, n\}$, with $s = 1$. We use the following three data structures.

For $i \geq 1$, we maintain a *price* π_i, giving the best distance from s to i that we have found so far.

R denotes a set of reachable vertices (those that have been added to the tree).

We use a set Q, which contains triples of the form (i, j, d). Such a triple means that we can reach j at cost d, by a path going from s to i (this part stays within R), followed by a direct link from i to j.

Initialization: Let $\pi_1 = 0$, and $R = \{1\}$. Make

$$Q = \{(1, j, d_{1j}) : 1 \leq j \leq n\}.$$

While $|R| < n$, repeat the following tree-growing step: Repeatedly identify elements $(i, j, d) \in Q$ that minimize d, and delete them until one is found with $j \neq R$. Then set $\pi_j := d$, $R := R \cup \{j\}$, and

$$Q := Q \cap \{(j, k, \pi_j + d_{jk}) : 1 \leq k \leq n\}.$$

We have expressed this algorithm in a form that is easy to analyze, without worrying about implementation details. You should be aware, however, that a good implementation will use a priority queue (heap) for Q, for which operations can be done in logarithmic time. Furthermore, things can be arranged so that Q contains only one "candidate" for each vertex in the complement of R. See, e.g. CLR pp. 527 ff.

Example.

Suppose the distance matrix is

$$\begin{pmatrix} 3 & 1 & 4 & 1 \\ 5 & 9 & 2 & 6 \\ 5 & 3 & 5 & 8 \\ 9 & 7 & 9 & 3 \end{pmatrix}$$

This corresponds to the network

Let's take a snapshot of the algorithm after initialization, and after each tree-growing step:

$R : \{1\}$ $\quad \pi : (0, \infty, \infty, \infty)$ $\quad Q :$ $(1,1,3), (1,2,1), (1,3,4), (1,4,1)$

$R : \{1,2\}$ $\quad \pi : (0, 1, \infty, \infty)$ $\quad Q :$ $(1,1,3), (1,3,4), (1,4,1),$
$(2,1,6), (2,2,10), (2,3,3), (2,4,7)$

$R : \{1,2,4\}$ $\quad \pi : (0, 1, \infty, 1)$ $\quad Q :$ $(1,1,3), (1,3,4),$
$(2,1,6), (2,2,10), (2,3,3), (2,4,7)$
$(4,1,10), (4,2,8), (4,3,10), (4,4,4)$

$R : \{1,2,4,3\}$ $\quad \pi : (0, 1, 4, 1)$ $\quad Q :$ $(1,3,4),$
$(2,1,6), (2,2,10), (2,4,7)$
$(4,1,10), (4,2,8), (4,3,10), (4,4,4)$
$(3,1,15), (3,2,11), (3,3,15), (3,4,12)$

So we get the tree

Average-case analysis of Dijkstra's algorithm

This analysis is reminiscent coupon collection.

The interesting question is, how many operations do we perform on Q?

Insertions: there are always n^2, since we put triples on the queue in batches of size n.

Deletions are more interesting.

Assume (for the analysis) that the matrix entries $d_{i,j}$ are i.i.d. samples from some continuous distribution. (Example: $U(0,1)$.)

Define a "success" to be the event that the minimizing triple (i,j,d) removed from Q has $j \notin R$. Since each success expands the tree, the algorithm stops when $n-1$ successes have occurred.

We now analyze the waiting time for a success, given that $|R| = k$. Note that

$$\Pr[\text{success}] = \sum_{i=1}^{n} \Pr[\text{success}|E_i]\Pr[E_i],$$

where E_i is the event that the minimizing triple has first coordinate i.

When we added the triples $(i,*,*)$ to Q, we put in n of them, of which k now have their first coordinate in R. A number, say t, of these, have been removed. Since we have no information about the relative rankings for tuples we have not yet used, we have

$$\Pr[\text{failure}|E_i] = \frac{k-t}{n-t} \leq \frac{k}{n}.$$

This estimate holds for all i, so we have

$$\Pr[\text{success}] = E[\Pr[\text{success}|E_i] \geq \frac{n-k}{n}.$$

This shows that the expected waiting time for the k-th success is $\leq n/(n-k)$. Sum this over k, and conclude

$$E[\#\text{ of deletes}] \leq \sum_{k=1}^{n-1} \frac{n}{n-k} \leq nH_n \sim n\log n.$$

Bottom line: We use relatively few deletions, compared to inserts. This suggests, among other things, that the idea of only inserting "viable" candidates – ones that beat the currently best known path to a given vertex – is a good idea.

Notes

See Noshita (J. Algorithms 1985); this may give a cleaner proof of this result.

Lecture 34

Topic du jour: Martingales and probabilistic counting

References:

[For martingales] Grimmett and Stirzaker, Probability and Random Processes, Chapter 12; Williams, Probability with Martingales, Chapter 10.

[For probabilistic counting] R. Morris, CACM, v. 21, 1978, pp. 840-842; W. Rosenkrantz, Stochastics, 1987.

Martingales

Let $X_0, X_1, \ldots$ be a sequence of random variables, all defined on the same underlying probability space. Assume $E|X_i| < \infty$ for all $i \geq 0$.

We call $\{X_i\}$ a *martingale* if

$$E[X_i|X_0, \ldots, X_{i-1}] = X_{i-1},$$

almost surely.

You should note the distinction between this and a Markov chain, which is a sequence for which the distribution X_i, given the "history" $X_0, \ldots, X_{i-1}$, is purely a function of X_{i-1}. For Markov chains the martingale property simplifies to

$$E[X_i|X_{i-1}] = X_{i-1}.$$

The phrase "almost surely" is there because both sides of the equality are random variables. For discrete problems we can (and will) ignore this fine point.

Its *difference sequence* is

$$d_i := X_i - X_{i-1}.$$

Intuitively, a martingale is supposed to capture the notion of a gambler's accumulated fortune while repeatedly playing a fair game. The difference sequence represents the gambler's winnings at each game.

Examples:

1. Let $Y_i = \pm 1$ for $i \geq 0$, independently and with equal probability. Let

$$S_n = \sum_{i=1}^{n} Y_i.$$

Note that this is a Markov chain. Then

$$E[S_n|S_{n-1}] = S_{n-1} + 1/2 - 1/2 = S_{n-1},$$

so the sequence S_n is a martingale.

2. [Casanova's martingale.] This is determined by the flips of a fair coin, as follows: We wager 1 dollar on the first flip. If we lose, we wager 2 dollars on the next, if that loses, 4 dollars on the next, and so on, doubling the stakes until we win. If Z_n denotes the accumulated win (or loss!) at the n-th trial, then Z_n is a martingale (since the net expected gain is 0 at each step). A win is certain. Unfortunately, you need unbounded money to play this game, since the mean loss you incur before the win is infinite. (Do you now see why casinos have bet limits?)
3. Let $T(X_1, \ldots, X_n)$ is a random variable. Then

$$Z_i := E[T|X_1, \ldots, X_i]$$

is a martingale with respect to $X_1, X_1, \ldots$, in the sense that

$$E[Z_i|X_1, \ldots, X_{i-1}] = Z_{i-1}.$$

You can think of Z_i as accumulating information about T, since

$$Z_0 = E[T] \qquad \text{(our first guess for } T\text{)}$$

whereas

$$Z_n = E[T|X_1, \ldots, X_n] = T(X_1, \ldots, X_n) \qquad \text{(the actual value of } T\text{)}$$

This is called the *Doob martingale process*. Many of the applications of martingales to computer science use it.

For any martingale X_i, we will have

$$E[X_i] = E[X_{i-1}] = \cdots = E[X_0].$$

This can be proved by induction on i. Intuitively, it says that we cannot expect to change our fortune by playing a sequence of fair games.

Example: Probabilistic Counting

Our goal is to count to n using $O(\log \log n)$ bits. Of course, this is impossible, so we rephrase the problem and ask for a procedure that gives us a good estimate of n (in some sense).

The following algorithm is due to R. Morris (CACM, 1978)

We use an integer variable b that starts at 1. For $t = 1, 2, \ldots$, we change b to

$$b' = \begin{cases} b+1, & \text{with probability } 2^{-b}; \\ b, & \text{with probability } 1 - 2^{-b}. \end{cases}$$

We now prove that $Z_t := 2^b - t$ is a martingale.

Since this is clearly Markov, it's enough to compute

$$\begin{aligned} E[Z_{t+1}|Z_t] = &\ E[Z_{t+1}|b] \\ = &\ 2^{b+1}2^{-b} + 2^b(1-2^{-b}) - (t+1) \\ = &\ 2^b - t = Z_t. \end{aligned}$$

Therefore,

$$E[2^b - t] = E[Z_t] = E[Z_0] = 2^1 - 0 = 2,$$

and we conclude that

$$E[2^b - 2] = t.$$

A statistician would say that $2^b - 2$ is an *unbiased estimator* for t.

Note that if $2^b \sim t$, then $\log_2 b \sim \log_2 \log_2 t$. We'd like to show that with high probability, $\log_2 b$ – which we think of as the number of bits in b – is never much larger than this.

We will use a standard result from probability theory called *Markov's inequality*.

Let W be a non-negative random variable with finite mean μ. Then

$$\Pr[W \geq a] \leq \mu/a.$$

You can prove this as an exercise.

Let's look at 4^b and see how this changes over time. We have

$$E[4^{b'}|b] = 4^{b+1}2^{-b} + 4^b(1-2^{-b}) = 4^b + 3 \cdot 2^b.$$

Therefore, on average, 4^b will increase by $3(t+2)$ at the t-th step. We guess, then, that the mean value of 4^b is $\Theta(t^2)$.

Let us prove this. Define

$$Y_t = 4^b - \frac{3}{2}t(t+3).$$

Then Y_t is constant in expectation, that is, $E[Y_{t+1}] = E[Y_t]$.

Proof: Compute

$$\begin{aligned} E[Y_{t+1} - Y_t|b] &= 4^{b'} - \frac{3}{2}(t+1)(t+4) - 4^{b'} + \frac{3}{2}t(t+3) \\ &= 3 \cdot 2^b - \frac{3}{2}(t+1)(t+4) + \frac{3}{2}t(t+3). \end{aligned}$$

So

$$E[Y_{t+1} - Y_t] = \frac{3}{2}((2t-4) - (t^2+5t+4) + (t^2+3t)) = 0.$$

Note. It does not follow that Y_t is a martingale, only that $E[Y_{t+1}] = E[Y_t]$. In general this is a weaker property.

By induction on t, then, $E[Y_t] = Y_0 = 4$. So

$$E[4^b] = \frac{3}{2}t(t+3) + 4.$$

Now we show that the chance that we need $c \log\log t$ bits for the counter goes down exponentially in c.

Using Markov's inequality, we have

$$\begin{aligned} \Pr[b \geq c\log_2 t] = &\ \Pr[4^b \geq t^{2c}] \\ &\leq \frac{(3/2)t(t+3)+4}{t^{2c}} \\ &\sim \frac{3}{2}t^{-2(c-1)}. \end{aligned}$$

From the work we've done, it follows that the variance of 2^b is $t(t+1)/2$. If we use Chebyshev's inequality, we can improve the constant in the result above from 3/2 to 1/2.

Notes

The term "martingale" goes back at least to the 18th century. Casanova's martingale (sometimes called *the* martingale, is related to the Petersburg game. See Grimmet and Stirzaker, p. 303 ff.

More examples of martingales:

Urn models for DNA computing (Bach, Condon, Glaser, Tanguay)

Document caching (= graph evolution of a random bipartite graph, look at the number of untouched Y-edges).

A card game martingale (good homework). Pay 1 dollar. Dealer shuffles a deck, turns over cards until you say "now". Win two dollars if the next card is red, nothing if it is black. The fraction of red cards after i draws is a martingale. Now use optional stopping.

Nice example from Valiant and Vazirani, NP is as Easy... (TCS around 1986). Start with a set S in $GF(2)^n$. Remove elements by intersecting with random hyperplanes. Then

$$Z_i := 2^i \times \text{size of set at time } i$$

is a martingale.

Lecture 35

Topic du jour: Optional stopping, multiplayer ruin problems

References:

Williams, loc. cit., p. 100.

E. Bach, Bounds for the expected duration of the monopolist game, IPL 101, 2007, pp. 86-92.

Martingales

Recall (from last lecture) the concept of martingale. The key property is

$$E[X_t|X_0,\ldots,X_{t-1}] = X_{t-1},$$

for all i. Often X_0 is constant and represents the initial state of a system.

Intuitively, a martingale tracks the fortune of a gambler who plays a seqence of fair games.

In expectation, martingales stay put, in the sense that (for all $t > 0$) $E[X_t] = E[X_0]$.

Stopping Times

Defn. A *stopping time* is a random variable (taking values in $\{0, 1, 2, \ldots, \infty\}$) where we can determine if $T = t$ by observing $X_0, \ldots, X_t$.

Example: I play a sequence of fair games, and quit after my first loss.

Non-example: I quit just before my first loss.

If T is a stopping time, is $E[X_T] = E[X_0]$?

No. Consider the Casanova martingale

$$X_t = \sum_{0\le i\le t} \pm 2^i.$$

If T is the time of the first win (+ sign),

$$E[X_T] = -(1 + 2 + 4 + \cdots 2^{T-1} + 2^T = 1,$$

whereas

$$E[X_0] = 0.$$

Constant expectation does hold in situations where the martingale and stopping time are "well behaved."

Theorem: Let $X_0, X_1, \ldots$ be a martingale for which $|X_i - X_{i-1}| \le B$ (uniformly bounded differences). Let T be a stopping time with $E[T] < \infty$. Then $E[X_T] = E[X_0]$.

This is a great tool for algorithm run time analysis, since we are often running a process, and waiting for some event to happen.

A Generalization of the Martingale Idea

Sometimes, it is useful to think a martingale as being "fed" by another source of randomness.

Defn: Let $X_0, X_1, \ldots$ be a sequence of random variables, all on the same sample space. A sequence $\{Z_t\}$ is a martingale with respect to $\{X_t\}$ if:

a) Z_t depends on $X_0, \ldots, X_t$;

b) $E[|Z_t|] < \infty$ for all t.

c) $E[Z_t | X_0, \ldots, X_{t-1}] = Z_{t-1}$.

It is useful to think of $X_0, \ldots, X_{t-1}$ as representing the information available (or visible) up to and including time $t-1$.

In the gambling metaphor, you can think of the X_t's as the cards, dice, coin flips, etc. that control the Z_t's.

All theorems we have mentioned (constant expectation, optional stopping) hold in this situation.

Multiplayer Ruin Games

The monopolist game (named by Vitányi and Watanabe) is a multi-player generalization of the gambler's ruin problem. It was invented as a possible mechanism for symmetry breaking during the self-organization of neurons.

The rules

k players have positive integral stakes $I_1, \ldots, I_k$.
While there are $m \geq 2$ active players:
Each contributes $1/m$ to the pot
A random winner takes the pot
Players who cannot pay the ante leave the game

At the end, there is a unique player ("the monopolist") with all the money.

This is not entirely obvious. One has to check that any fortune below $1/m$ is actually 0.

Let T be the time at which the monopolist emerges. This is a stopping time, and $E[T] < \infty$. (Idea of proof: any player can be bankrupted by losing some fixed number of times in a row. This event has positive probability, so the expected waiting time for it is finite.)

Probability to Win

Let $X_i(t)$ be the i-th players fortune at time t. If he is active, he loses $1/m$ with probability $(m-1)/m$, and wins $(m-1)/m$ with probability $1/m$. So his expected gain is zero, making

$$E[X_i(t) - X_i(t-1)] = 0, \qquad t \geq 1.$$

So $\{X_i(t)\}$ is a martingale.

We have

$$X_i(T) = \begin{cases} 0, & \text{if } i \text{ loses the game;} \\ \sum_j I_j = S, & \text{if } i \text{ wins.} \end{cases}$$

Now use optional stopping.

$$E[X_i(T)] = S \cdot \Pr[\ i \text{ wins }] = E[X_i(0)] = I_i.$$

So

$$\Pr[\ i \text{ wins }] = \frac{I_i}{S}.$$

We proved this for $k = 2$ using recurrence relations.

It is tempting to think about an economic interpretation. Even if the individual games are fair, the probability that a given player wins is the fraction of total wealth that he starts with. If I enter the business world with \$1K my career is rather different than if I enter it with \$1B.

Duration of the Game

It is convenient to think of a "fortune vector" $X = (X_1, \ldots, X_k)$. We will also take our source of randomness to be the (vector) random process $X(0), X(1), \ldots$. (Knowing this is the same as knowing the winner for each round.)

Let $||X(t)||^2 = \sum_{i=1}^k X_i(t)^2$. Then, if δ_i denotes the change in player i's fortune,

$$\begin{aligned} ||X(t+1)||^2 &= \sum_{i=1}^k (X_i(t) + \delta_i)^2 \\ &= \sum_{i=1}^k X_i(T)^2 + 2\sum_{i=1}^k \delta_i X_i(t) + \sum_{i=1}^k \delta_i^2. \end{aligned}$$

So

$$E\left[||X(t+1)||^2 | X(t)\right] = ||X(t)||^2 + 2\sum_{i=1}^k E[\delta_i X_i(t) | X(t)] + \sum_{i=1}^k E[\delta_i^2 | X(t)].$$

The i-th term of the first sum equals

$$X_i(t) E[\delta_i | X(t)] = 0,$$

since the game is fair from i's perspective.

The second sum requires some computation. If i is one of the m active players,

$$\delta_i = \begin{cases} -\frac{1}{m}, & \text{with probability } \frac{m-1}{m}; \\ \frac{m-1}{m}, & \text{with probability } \frac{1}{m}. \end{cases}$$

So

$$E[\delta_i^2] = \frac{1}{m^2} \cdot \frac{m-1}{m} + \frac{(m-1)^2}{m^2} \cdot \frac{1}{m},$$

making

$$\sum_{i=1}^{k} E[\delta_i^2] = \frac{m-1}{m^2} + \frac{(m-1)^2}{m^2} = \frac{m-1}{m}.$$

(For inactive players, $\delta_i = 0$.)

So we can make a martingale:

$$Z(t) = ||X(t)||^2 - \sum_{1 \le u \le t} \frac{m(u)-1}{m(u)}.$$

This has bounded differences (the total amount of money in the game is finite). Applying optional stopping,

$$E[Z(T)] = Z(0) = \sum_{i=1}^{k} I_i^2.$$

For this lecture, we make the simplifying assumption that all initial stakes are equal, so $I_i = I$ for all i. Then

$$Z(0) = kI^2,$$

and

$$Z(T) = (kI)^2 - \sum_{1 \le u \le T} \frac{m(u)-1}{m(u)}.$$

Since these two quantities are equal,

$$\frac{T}{2} \le \sum_{1 \le u \le T} \frac{m(u)-1}{m(u)} = (k^2 - k)I^2 \le T.$$

(The outer inequalities follow from $1/2 \le (m-1)/m \le 1$.) Taking expectations, we get

$$\frac{E[T]}{2} \le k(k-1)I^2 \le E[T],$$

so $E[T] = \Theta(k^2 I^2)$.

According to the last result, the expected game duration is upper bounded by a constant times the square of the total initial stakes. This remains true for unequal stakes (provided they are integral).

Notes

For integral stakes, the expected game duration can be computed exactly, when there are 3 players.

Ross has studied a similar game, in which the ante is constant irrespective of the number of players. See his paper, The multiple-player ante one game, Prob. Eng. Inform. Sci. 25, 2011, 343-353.

Lecture 36

Topic du jour: Geometric TSP

References: Karp and Steele (in E. L. Lawler et al., ed., The Traveling Salesman Problem).

The traveling salesman problem

This is the archetypal hard (NP-complete) combinatorial optimization problem. We consider a geometric version.

Let $X_1, \ldots, X_n$ be n points in the unit square $[0,1]^2$. We consider tours through these points that visit each point exactly once. The problem is to find one that is cheapest (minimizes the total distance between points on the tour).

Beardwood, Halton, and Hammersley (1959) studied an average-case version of this problem. Among other things, they showed that if $X_1, \ldots, X_n$ are i.i.d. samples from the uniform distribution on the unit square, then there is a positive constant β such that

$$E[\text{ cost of optimal tour }] \sim \beta\sqrt{n}.$$

A precise value for β is not known.

They proved similar results for higher-dimensional cubes, which we don't need here.

We will prove a weaker version of this and show that $E[$ cost of optimal tour $] = \Theta(\sqrt{n})$.

A heuristic argument

Before doing anything rigorous it is worthwhile to see why $\sqrt{n}$ is a plausible order of magnitude.

Suppose the points formed a $\sqrt{n} \times \sqrt{n}$ regular grid, exactly $\sqrt{n}$ apart horizontally and vertically. (Part of the beauty of heuristics is that we can pretend that $\sqrt{n}$ is always an integer!)

If we "snake" through these points, the total cost is about $\sqrt{n}$.

Conversely, we can surround each point by its own circle of radius $r = 1/(2\sqrt{n})$. The path must enter the circle, go to the point, and leave again. This incurs a cost of at least $2rn = \sqrt{n}$.

It should not be missed that we have found a tour for this case whose cost is optimal up to low order terms.

A lower bound on the expected value of the best tour

Let $m = \lfloor \sqrt{n} \rfloor$. Divide the square into m^2 square cells whose length and width equals $h = 1/m$.

Further subdivide each cell into nine congruent square subcells. We will call a cell "good" if it has exactly one point, which lies in the innermost subcell. Note that in this case, it costs at least $2h/3$ to connect this point to the rest of the tour.

Given that a cell has exactly one point, it is good with probability 1/9.

The number of points in a cell has a binomial distribution, so by our usual trick,

$$E[\text{ \# of singly occupied cells }] = n\Pr[\text{ \# a cell is singly occupied }]$$

$$= n \times \binom{n}{1} n^{-1}(1-1/n)^{n-1} \sim e^{-1}n.$$

The optimal tour therefore has expected cost at least

$$e^{-1}n \times 1/9 \times 2h/3 \sim 0.027250...\sqrt{n}.$$

An upper bound on the expected value

We will provide an algorithm to construct a tour of cost $O(\sqrt{n})$ through any set of n points.

Suppose we have n points in the unit square.

Let $m = \lceil \sqrt{n} \rceil$ and $h = 1/m$. Divide the square into m strips of unit width and height h. For each strip, we use one corner (upper left for "odd" strips and upper right for "even" ones) as an input and its opposite corner as an output. This adds $m+1$ new points to the tour.

Now sort the points in each strip by x-coordinate and use this to rout the tour through the strips:

Let the total cost be C. Then

$$C \leq \text{ total } \Delta x + \text{ total } \Delta y + \sqrt{2}.$$

(We get the extra $\sqrt{2}$ from the last edge, which at worst connects opposite corners.) We have

$$\text{total } \Delta x \leq [\ \#\ \text{of strips}\ ] = m \leq \sqrt{n} + 1$$

and

$$\text{total } \Delta y \leq (n+m)h \leq \sqrt{n} + 1.$$

So the total cost is bounded by $2\sqrt{n} + (2 + \sqrt{2})$.

This is an extremely fast procedure: it cost is dominated by sorting so it is $O(n \log n)$.

Notes

The following papers are relevant. R. M. Karp, The probabilistic analysis of some new combinatorial search algorithms, in J. F. Traub, ed., Algorithms and Complexity: New Directions and Recent Results, Academic Press, 1976; R. M. Karp, Math. of O. R. 2, 209-224.

Lecture 37

Topic du jour: Concentration laws for martingales; application to law of large numbers for geometric TSP.

References:

[For TSP] Rhee and Tallagrand, Math. of O. R. 1987; Karp and Steele (in E. L. Lawler et al., ed., The Traveling Salesman Problem). We follow Grimmett and Stirzaker, pp. 450-452.

[For martingale concentration] Grimmett and Stirzaker, pp. 448-449. See Williams, p. 237 for a stronger result.

[For Doob's martingale] Grimmett and Stirzaker, pp. 456-457. Williams, p. 96.

The Azuma-Hoeffding inequality

One of the nice features of martingales is that they tend to stay put.

Let $Z_0, Z_1, \ldots, Z_n$ be a martingale. Let $K_1, K_2, \ldots$ be a sequence bounding its differences d_i, that is, we have

$$|Z_i - Z_{i-1}| \leq K_i$$

for all i. Let $K = \sum_{i=1}^{n} K_i^2$. Then for any $\lambda > 0$,

$$\Pr[\exists i |Z_i - Z_0| \geq \lambda] \leq 2\exp(-\lambda^2/(2K)).$$

For a proof, see D. Williams, Probability with Martingales, p. 237.

We will prove a weaker version of this inequality, without the $\exists i$ in the predicate. It should be noted that in practice, the right hand side is often exponentially small, making the weaker version just as useful as the stronger version.

Our proof followed Grimmett and Stirzaker, Probability and Random Processes, pp. 448-449.

Here are some examples of this inequality in action.

Let $X_1, \ldots, X_n$ be i.i.d. Bernoulli trials with success probability p. The normalized number of successes in k trials,

$$Z_k := \sum_{i=1}^{k} X_i - kp$$

forms a martingale with mean 0. We note that $|Z_k - Z_{k-1}| \leq 1$. Therefore (taking $\lambda = n\epsilon$), we have

$$\Pr\left[\left|\frac{S_n}{n} - p\right| \geq \epsilon\right] \leq 2e^{-n\epsilon^2/2}.$$

This is a version of the law of large numbers with an explicit bound on the tails.

What's the maximum deviation from 0 in a symmetric random walk? The successive positions $S_0, S_1, \ldots$ form a martingale with differences bounded by 1. The inequality gives

$$E[|\max_{i \le n}\{S_i\}|] = \int_0^\infty \Pr[\max > \lambda] d\lambda \le 2\int_0^\infty e^{-\lambda^2/2n} = \sqrt{2\pi n}.$$

On the other hand, we expect a deviation comparable to $\sqrt{n}$ by, say, the central limit theorem. So the maximum deviation is $\Theta(\sqrt{n})$.

For the Casanova martingale, $d_i = 2^{i-1}$, so $\sum_{i=1}^n d_i^2 = (4^n - 1)/3$. This leads to the uninformative bound

$$\Pr[|Z_n| \ge \lambda] \le 2e^{-3\lambda^2/2^{2n+1}}.$$

Review of TSP

We place n points uniformly and independently on the unit square.

There is no advantage to connecting points other than by line segments, and there are $(n-1)!$ such tours through these points. One of these must have minimal cost.

We proved last time that the optimal tour has expected cost $\Theta(\sqrt{n})$. Define β_n to be this expected cost, divided by $\sqrt{n}$. It is known that the limit of β_n is some mysterious constant β.

Today we wish to prove that if the random variable T_n denotes the cost of the optimal tour (through n random points), then for every $\epsilon > 0$,

$$\Pr[|T_n/\sqrt{n} - \beta_n| > \epsilon] \to 0.$$

You can think of this as analogous to the law of large numbers.

A weak law of large numbers for TSP

Let T be the cost of the optimal tour.

We use the martingale

$$Z_0 = E[T], \ldots, Z_i = E[T|X_0, \ldots, X_i], \ldots, Z_n = E[T|X_0, \ldots, X_n].$$

Note that these are all functions of the points we choose, and that $Z_n = T$.

Any time T is a "nice" function of $X_0, \ldots, X_n$, we can do this. It is often called *Doob's martingale*.

The idea is that we are learning more and more about the value T as we reveal more of its inputs.

Let $T(i)$ denote the cost of the mimimum tour visiting all points but X_i.

The key observation is this. For all i, we have

$$T(i) \le T \le T(i) + 2U_i, \qquad (*)$$

where

$$U_i = \begin{cases} \min\{d(X_i, X_j) : i < j \le n\}, & \text{if } 1 \le i < n; \\ \sqrt{2}, & \text{if } i = n. \end{cases}$$

To prove the lower bound, note that T is at least the cost of an optimal tour with X_i spliced out. This gives some tour through $X_1, \ldots, X_{i-1}, X_{i+1}, \ldots X_n$, whose cost is not smaller than the optimum, $T(i)$.

For the upper bound, find an optimal tour through $X_1, \ldots, X_{i-1}, X_{i+1}, \ldots X_n$ (in some order). If we splice in X_i, we get a tour through all n points whose cost is $\le T(i) + 2\sqrt{2}$, by the triangle inequality. This expression is therefore an upper bound on the cost of the optimum tour.

However, if $i < n$ we can do better. Let X_j be the closest point to X_i among $X_{i+1}, \ldots, X_n$. Let X_k follow X_j on the tour. If we replace the edge $X_j \to X_k$ by the edge $X_j \to X_i \to X_k$, the cost goes up by at most $2U_i$. (By our choice of j, $d(X_j, X_i) = U_i$. By the triangle inequality, $d(X_i, X_k) \le U_i + d(X_j, X_k)$.) So

$$T \le T(i) + 2U_i,$$

as needed.

Take conditional expectations of (*), to get

$$E[T(i)|X_1, \ldots, X_i] \le Z_i \le E[T(i)|X_1, \ldots, X_i] + 2E[U_i|X_1, \ldots, X_i]$$

and (after flipping signs around)

$$-E[T(i)|X_1, \ldots, X_{i-1}] + 2E[U_i|X_1, \ldots, X_i]. \le -Z_{i-1} \le -E[T(i)|X_1, \ldots, X_{i-1}] \le$$

Since $T(i)$ is insensitive to the value of X_i, we have

$$E[T(i)|X_1, \ldots, X_i] = E[T(i)|X_1, \ldots, X_{i-1}],$$

so

$$-2E[U_i] \le Z_i - Z_{i-1} \le 2E[U_i|X_i]. \qquad (**)$$

Let us now find a uniform bound for $E[U_i|X_i]$ when $i < n$.

Lemma: if P is a point in the unit square, then there is a positive constant c ($= \pi/32$) with the following property. For $0 \le r \le \sqrt{2}$, the disk of radius r around P intersects the unit square in a region of area $\ge cr^2$.

Proof: Divide $[0,1]^2$ into four congruent squares. Without loss of generality P lies in the upper right one. If $0 \le r \le 1/2$, the quarter circle of radius r to the "southwest" of P lies within $[0,1]^2$, so the intersection area is $\ge \pi r^2/4$. If $1/2 \le r \le \sqrt{2}$, the intersection still contains the quarter circle of radius $1/2$, so its area is

$$\ge \frac{\pi}{16} = \frac{\pi(\sqrt{2})^2}{32} \ge \frac{\pi r^2}{32}.$$

Using the lemma,

$$\Pr[d(X_i, X_j) \le r | X_i] \ge cr^2$$

so

$$\Pr[d(X_i, X_j) > r | X_i] \ge 1 - cr^2.$$

Therefore

$$\Pr[U_i > r | X_i] = \Pr[\forall j > i \; d(X_i, X_j) > r] \le (1 - cr^2)^{n-i} \le \exp(-c(n-i)r^2).$$

This gives

$$E[U_i | X_i] \le \int_0^\infty \exp(-c(n-i)r^2) dr = \frac{1}{2}\sqrt{\frac{\pi}{c}}(n-i)^{-1/2}.$$

The same bound holds for $E[U_i]$ as well, by the law $E[U] = E[E[U|X]]$.

Let $d_i = Z_i - Z_{i-1}$. From (**), we now easily obtain

$$\sum_{i=1}^{n} d_i^2 \le \frac{\pi}{c} H_{n-1} + 8,$$

which is bounded by $D := A \log n + B$ for suitable constants A and B. (From $c = \pi/32$ we can obtain $A = 32$, $B = 40$.) Plugging this into the Azuma-Hoeffding inequality (see last lecture), we find that

$$\Pr[|T/\sqrt{n} - \beta_n| > \epsilon] = \Pr[|Z_n - Z_0| > \epsilon\sqrt{n}] \le 2 \exp\left(-\frac{\epsilon^2 n}{2(A \log n + B)}\right).$$

As required, for fixed $\epsilon > 0$, this goes to 0.

Lecture 38

Topic du jour: The second moment method.

Reference: Lectures by R.M. Karp, 1987.

What's this about?

This is useful when we have a random variable

$$Y = \sum_{i=1}^{n} X_i,$$

and the X_i's are indicators of events.

To estimate $\Pr[Y \geq 1]$, we can use the inequality

$$\Pr[\exists i \ X_i = 1] \leq nE[X_i].$$

How might we estimate $\Pr[Y = 0]$?

We have previously used inclusion-exclusion (the combinatorial sieve).

The second moment method provides an alternative and requires knowledge only of the first two moments of Y.

Chebyshev's inequality and the Second Moment Estimate

Let Y be a random variable whose first two moments exist. Then

$$\Pr[|Y - EY] \geq t] = \Pr[(Y - EY)^2 \geq t^2] \leq \frac{E[(Y - EY)^2]}{t^2} = \frac{\sigma^2(Y)}{t^2}.$$

Now assume that Y takes only non-negative integer values. Applying Chebyshev's inequality with $t = EY$ we get

$$\Pr[Y = 0] \leq \Pr[|Y - EY| \geq EY] \leq \frac{\sigma^2(Y)}{(EY)^2}.$$

For some applications it is convenient to rewrite the bound as

$$\frac{E(Y^2)}{(EY)^2} - 1.$$

L. Shepp (See Karp et al., J. Appl. Prob 1986) has observed that we can sharpen this to

$$\Pr[Y = 0] \leq \frac{\sigma^2(Y)}{E(Y^2)}.$$

There is nothing holy about Chebyshev's inequality. You can use other concentration inequalities.

Example 1: binomial distribution.

Let Y be the number of successes in n trials, where each trial has (independently) the success probability p. Put $q = 1 - p$ as usual. Then

$$\Pr[Y = 0] = q^n \leq \frac{npq}{(np)^2} = \frac{q}{np}.$$

Note that the true probability goes down exponentially when $q < 1$, whereas the estimate is $O(1/n)$.

Example 2: pseudo-random search.

Let the universe be $U = \{0, 1, \ldots, p-1\}$, where p is prime.

Suppose S is a subset of U, of size $\geq p/2$. We assume we can recognize an element of S when we find it. Our search process will generate a sequence

$$T_0, T_1, \ldots \in U$$

by a pseudo-random process and stop when $T_i \in S$.

To make the sequence we choose two seeds $\alpha, \beta \in U$ (uniformly and independently) and then let

$$T_i = \alpha i + \beta \bmod p.$$

(That is, we choose a random starting place and then use a fixed, but randomly chosen offset to get each new iterate.) What is the probability that none of $T_1, \ldots, T_n$ belong to S?

Let $X_i = 1$ if $T_i \in S$ and 0 otherwise. Put $Y = \sum_{i=1}^{n} X_i$. So we wish to estimate

$$\Pr[Y = 0].$$

If the T_i were independent, we could use the binomial result we proved above, and conclude that

$$\Pr[Y = 0] \leq 2/n.$$

It turns out that the T_i are pairwise independent, which is enough to make this argument go through. Let's check this.

Choose $a, b \in U$, and let $0 \leq i < j < p$. If

$$\alpha i + \beta = a$$
$$\alpha j + \beta = b$$

then

$$\alpha(j-i) = b - a.$$

It can be shown that the mapping

$$x \mapsto rx \bmod p$$

is 1-1 on U if $0 \le r < p$. So there is exactly one α that does the job, and from it one can find β. Therefore

$$\Pr[T_i = a \text{ and } T_j = b] = p^{-2} = \Pr[T_i = a]\Pr[T_j = b].$$

One of the standard variance formulas is

$$\sigma^2(Y) = \sum_{i=1}^{n} \sigma^2(X_i) + 2\sum_{i<j} \mathrm{Cov}(X_i, X_j).$$

Since each pair X_i, X_j is independent, we will get the same result as if all the X_i were independent. This gives

$$\sigma^2(Y) = n\pi(1-\pi)$$

where π is the probability that $X_i = 1$.

We assumed that $\pi \ge 1/2$, so the second moment estimate gives

$$\Pr[Y = 0] \le \frac{1-\pi}{n\pi} \le \frac{1}{n}.$$

Example 3: when is a random mapping onto?

This is a static version of the coupon collector's problem. We throw $N = \alpha n$ balls into n bins. Clearly, every bin can be occupied only when $\alpha \ge 1$. Since the waiting time to get all bins occupied is $\sim n \log n$, we expect that the probability that all bins are occupied to go to zero with n. The second moment method allows us to estimate the rate at which this happens.

Let X_i be 1 if the i-th bin is empty, and 0 otherwise. Put $Y = \sum X_i$.

Since $EX_i = (1 - 1/n)^N$, we have

$$E[Y] = n(1 - 1/n)^N.$$

.

To compute the second moment of Y, we expand Y^2:

$$Y^2 = \sum_{i=1}^{n} X_i^2 + 2 \sum_{1 \le i < j \le n} X_i X_j.$$

We note that

$$E[X_i^2] = E[X_i] = (1 - 1/n)^N,$$

and

$$E[X_i X_j] = \Pr[\text{bins } i \text{ and } j \text{ empty}] = (1 - 2/n)^N.$$

From this,

$$E[Y^2] = n(1 - 1/n)^N + n(n-1)(1 - 2/n)^N.$$

Plug these into the estimate and get

$$\Pr[Y = 0] \leq \frac{n(1 - 1/n)^N + n(n-1)(1 - 1/n)^N}{n^2(1 - 1/n)^N} - 1.$$

If $N = \alpha n$ and $n \to \infty$, our estimate is asymptotic to

$$\frac{e^\alpha}{n}.$$

Lecture 39

Topic du jour: Erdős's probabilistic method.

References: M. Molloy, The Probabilistic Method, in M. Habib et al., Probabilistic Methods for Algorithmic Discrete Mathematics, pp. 1-30. See also J. Spencer, Ten Lectures on the Probabilistic Method. For prime testing see J. D. Dixon, AMM 1984.

The probabilistic method

Often we are interested in proving that an object with certain properties exists. The traditional ways for doing this are:

1. Give an algorithmic construction for the object.
2. Show that the non-existence of the object is contradictory. (This was controversial in Hilbert's time but now routine.)

In 1947, Paul Erdős proved a result in Ramsey theory by the following ingenious method:

3. Show that for a randomly chosen object, of the appropriate type, the probability that is has the property is nonzero. (Of course this requires a clever choice of the probability space.)

This method is by now standard. (See Spencer for many examples.) We will give some examples from logic and number theory.

Satisfiability

We consider Boolean formulas in conjunctive form, such as

$$(x_1 \vee x_2 \vee x_3) \wedge (\overline{x_1} \vee \overline{x_2} \vee x_3) \wedge (x_1 \vee x_2 \vee \overline{x_3})$$

The formulas inside the parentheses are called *clauses*.

Each clause is an "or" of one or more *literals*.

A literal is a variable or its complement.

The formula is satisfiable if some assignment to the x_i makes it evaluate to 1. For example, here we can take $x_1 = 0$, $x_2 = 1$, and x_3 to be anything.

We call a Boolean formula (in conjunctive form) a k-formula if each clause has exactly k literals, using distinct variables. Thus our example is a 3-formula.

How does the complexity of satisfiability depend on k?

Satisfiability of 1-formulas is trivial to decide.

There are polynomial time algorithms (e.g. resolution) to decide if a 2-formula is satisfiable.

Satisfiability of 3-formulas is an NP-complete problem. This means there is no fast algorithm for this, nor is it likely there ever will be.

We now wish to prove that, roughly, that a k-formula is satisfiable if it is short (has few clauses) or does not have much dependence between the clauses (in a sense we will define).

Satisfiability of short formulas

Theorem: any k-formula with $< 2^k$ clauses is satisfiable.

We may as well assume the clauses, considered as sets of literals, are distinct.

The proof involves considering the effect of a random assignment. Let X_i be 1 if the i-th clause is satisfied, 0 of not. Then

$$Y = \sum_i X_i$$

counts the number of unsatisfied clauses. Since

$$E[x_i] = 2^{-k},$$

we have

$$E[Y] = \sum_i E[X_i] < 2^k \cdot 2^{-k} = 1.$$

Now observe that

$$\Pr[\text{ there is an unsatisfied clause }] \leq E[\text{ \# of unsatisfied clauses }] < 1,$$

so the probability that all clauses are satisfied is positive.

We conclude that there is some satisfying assignment.

It is notable that our proof did not construct a satisfying assignment for us, it just verified the existence of one.

With a similar argument one can prove the following result. Let n_i be the number of literals in the i-th clause. If

$$\sum_i 2^{-n_i} < 1,$$

the formula is satisfiable.

Corollary: Call a clause of an n-variable formula "large" if it has $\geq 2^{n/2}$ literals. Formulas composed of large clauses can be tested for satisfiability in polynomial time.

If the formula has $< 2^{n/2}$ clauses it is satisfiable. Otherwise, the possible number of assignments is 2^n, and this is at most the square of the input length. So it can be tested in polynomial time.

There is nothing magic about 1/2, which could be replaced by any $\epsilon > 0$.

The Lovasz Local Lemma

Often we wish to use the probabilistic method

Let $E_1, \ldots, E_n$ be events. An undirected graph with vertices $E_1, \ldots, E_n$ is a dependency graph for these events if for every i, the set of random variables

$$E_i \cup \{E_j : E_j \text{ is not a neigbor of } E_i\}$$

is mutually independent.

Every set has at least one dependency graph: the complete graph K_n.

Lemma: Let $E_1, \ldots, E_n$ be events of probability at most $p < 1$, with a dependency graph of degree $\leq d$. Then if

$$4pd < 1,$$

we have Pr[no E_i occurs] > 0.

The proof is not hard but we will not do it here. A proof can be found in Spencer's book.

As a check, suppose that the events are independent. Then

$$\Pr[\text{ no } E_i \text{ occurs }] \geq (1-p)^n > 0.$$

Satisfiability problems with weak dependence.

Theorem: Let f be a k-formula such that each variable occurs in $\leq 2^{k-2}/k$ clauses. Then f is satisfiable.

To prove this, we again select a random assignment. Let E_i be the event that the i-th clause is false. We link E_i to E_j if clauses i and j share a variable. This gives a dependency graph, whose degree d is at most

$$k\left(2^{k-2}/k - 1\right) < 2^{k-2}.$$

Since we have a k-formula the probability a clause is not satisfied is $p = 2^{-k}$.

The lemma gives $4pd < 1$, so there is a positive probability that our assignment will satisfy the formula.

We conclude as before that the formula is satisfiable.

Prime testing [this wasn't done this time around]

Background: The decision problem for primality (given n, is it prime?) is not known to be solvable in polynomial time. You might think this is so because the smallest Boolean circuit to test numbers (of a fixed size) for primality is necessarily large. We will show that this is not so.

Miller-Rabin prime test (a randomized algorithm).

Let $n = 2^\nu m + 1$, where m is odd. (We allow $\nu = 0$.)
Choose a, $1 \le a < n$, at random.
Compute $a^m, a^{2m}, \ldots, a^{n-1} \bmod n$.
If the sequence ends with 1 and -1 precedes the first 1, say "prime."
(We allow all 1's.)
Otherwise, say "composite."

This algorithm gives a proof of compositeness but only statistical evidence of primality, albeit with high probability.

We call a a *witness* for n if n is composite and the algorithm says so. If n is composite and the algorithm says "prime," we call a a *liar*.

It can be shown that the error probability is $\le 1/4$ for odd n and $\le 1/2$ for even n.

The test uses $O(\log n)$ multiplications mod n if you implement exponentiation by repeated squaring. (Look at any algorithms book if you don't know how to do this.)

Dixon's theorem

Question: Is there a "universal" set of bases a that, taken together, give correct answers for every n?

This is true in the trivial sense (make the set be all bases).

If a famous conjecture of number theory (Extended Riemann hypothesis) is true, then the bases $\le 2(\log n)^2$ work for all $m \le n$.

Dixon proved that there is a small universal set of bases, We will prove this, as an example of the probabilistic method.

Theorem (Dixon): There is a set S of bases that give correct results for every n satisfying $N/2 < n \le N$, such that

$$|S| \sim 2 \log_2 N.$$

Proof: We only have to worry about odd values of n, since $a = 2$ works for every even n. You can verify that if n is odd, a is a liar for n iff $-a$ is.

We will choose our bases from $1, 2, \ldots, N/2$. By the above symmetry property, the chance that a random base a works for n is $\ge 3/8$. (It is $\ge 3/4$ for the bases $< n/2$, and ≥ 0 for the others.)

Therefore

$$\Pr[a \text{ is a liar for } n] \le \frac{5}{8} < 2^{-1/2}.$$

If we choose k random bases (repetitions are allowed),

$$\Pr[a_1, \ldots, a_k \text{ are all liars for } n] \le 2^{-k/2}.$$

So

$$\Pr[\exists n \ (\ a_1, \ldots, a_k \text{ are all liars for } n\ )\ ] \le \frac{N}{2} 2^{-k/2}.$$

This is less than 1 as soon as $k > 2\log_2 N - 2$.

Notes

One way to summarize the Erdos argument is to say that we prove a set is nonempty by proving that it is large. Here, "large" is in the sense of probability (or, measure). There are several other variations on this idea; see, e.g. W. Rudin, Real and Complex Analysis, Section 8.13.

Lecture 40

Topic du jour: Markov Chains, Monte Carlo

References: Feller I, XV.1–XV.7 (for Markov chains); I. M. Sobol, The Monte Carlo Method (for classic Monte Carlo); M. Jerrum, Mathematical Foundations of the Markov Chain Monte Carlo Method, in Habib et al, pp. 116-165.

What's this about?

Often we want to sample from a distribution that is hard to draw from exactly. One way of doing this is to cook up a stochastic process $X_1, X_2, \ldots$ that is easy to implement, with the property that the distribution of X_i converges to the desired one. We run the process for n steps and then use X_n as our sample.

This is usually done by a process that walks through a finite state space. This has acquired the name "Markov Chain Monte Carlo." To get started we will explain what these two phrases mean.

The Monte Carlo Method

This was popularized in the 1950's as a way to estimate integrals that are difficult to compute exactly.

Let $f; [0,1]^n \to \mathbf{R}$ be some function we can evaluate at any desired point. How might we compute

$$\int_{[0,1]^n} f(x_1, \ldots, x_n) dx_1 \ldots dx_n?$$

In calculus one is taught to do such integrals by reducing them to iterated one-dimensional integrals. There are two problems with this.

The intermediate integrals may be hard to do exactly. This holds in spades if we are integrating not over a rectangle but over some oddly shaped region.

If you are trying to do it numerically, you need one $n-1$ dimensional integral for each sample point at the top level. Each of those must be evaluated by computing $n-2$ dimensional integrals, and so on. The details of this are not pleasant to contemplate.

Here is an alternative approach. Let ξ be a random point from $[0,1]^n$. Then $f(\xi)$ is a random variable whose expected value is the integral. If the variance of this random variable is small, we can use the following approximation:

$$\int f(x) \approx \frac{1}{N} \sum_{i=1}^{N} f(\xi_i).$$

Typically the error is of order $N^{-1/2}$.

Usually the variance is not known. (Remember the whole reason you are doing this is because you don't know the expected value. Now you want the variance?)

Markov Chains

Definition: A sequence of random variables $X_1, X_2, \ldots$ is called a *Markov Chain* if the distribution of X_i, given $X_0, \ldots, X_{i-1}$, is a function only of X_{i-1}.

In particular, it does not depend on the future.

Make sure you understand the difference between this and a martingale. The two concepts are related but not the same, and neither implies the other.

Concrete realizations of Markov chains

Let Ω be a finite state space. (Infinite state spaces are allowed in the general theory, but we will not need them.)

We also have transition probability $p_{x,y}$ for each pair of states. For our applications we will assume that these probabilities do not change with time.

The process evolves as follows. Choose a start state $x_0 \in \Omega$. (This can be done by a random process or deterministically.) Now, for $t = 0, 1, 2, ...$ we do the following. Suppose that $X_t = x$. Select y with probability $p_{x,y}$ and set $X_{t+1} = y$.

We have defined the process so that for all $t \geq 0$

$$\Pr[X_{t+1} = y | X_t = x] = p_{x,y}.$$

An example.

Let $\Omega = \{A, B\}$.

The transition probabilities can be arranged in the following matrix:

	A	B
A	0.5	0.5
B	0.1	0.9

If we are in state A, we go to either A or B with equal probability.

If we are in state B, we stay in B with probability 0.9, and go to A with probability 0.1.

Each row of our matrix adds up to 1. Matrices with this property are called *stochastic*. Thus, to give a Markov chain (finite state space, unchanging transition rules) is the same as to give a stochastic matrix.

One can also specify the chain by a diagram, similar to those drawn in finite automata theory:

If we retain only the edges that have positive probability and ignore the numerical values, we obtain a directed graph. This is called the underlying graph of the Markov chain.

Definition: A finite-state Markov chain is called *ergodic* if: 1) the underlying graph is strongly connected (this means that there is a path between any pair of states), and 2) the cycle lengths have greatest common divisor 1 (this guarantees that the behavior of the chain will not be periodic).

When Ω is infinite ergodic chains are required to satisfy more conditions. We will ignore this fine point.

It can be shown that if the chain is ergodic, there is a unique probability distribution π, such that if P is the transition matrix,

$$\pi P = \pi.$$

Furthermore, for any initial distribution on X_0. the distribution of X_i converges to π. This is called the *stationary distribution* of the Markov chain.

What is the stationary distribution for our example?

Clearly 1) and 2) hold.

With

$$P = \begin{pmatrix} 1/2 & 1/2 \\ 1/10 & 9/10 \end{pmatrix}$$

we want to solve $\pi P = P$, which is the same as $\pi(P - I) = 0$. Writing out the equations we have

$$-1/2\pi_1 + 1/2\pi_2 = 0$$
$$1/10\pi_1 - 1/10\pi_2 = 0$$

These are dependent so we need one more equation. Since π is a probability distribution we must have

$$\pi_1 + \pi_2 = 1.$$

The solution is

$$\pi_1 = 1/6, \pi_2 = 5/6.$$

This tells us that if we run the chain for a long time and see what state it is in, it will be A with probability about 17% and B with probability about 83%. This is reasonable since the state B is "sticky" – once we land there we are likely to stay there a while.

A quick and dirty method for computing stationary distributions

Since P^n is the transition matrix for n steps, we can compute the stationary distribution by successively squaring P.

If we do this for our example we get

$$P = \begin{pmatrix} .5 & .5 \\ .1 & .9 \end{pmatrix}, P^2 = \begin{pmatrix} .30 & .70 \\ .14 & .86 \end{pmatrix}, P^4 = \begin{pmatrix} .1880 & .8120 \\ .1624 & .8376 \end{pmatrix},$$

$$P^8 = \begin{pmatrix} .16721 & .83279 \\ .16656 & .83344 \end{pmatrix}, P^{16} = \begin{pmatrix} .16667 & .83333 \\ .16667 & .83333 \end{pmatrix}.$$

Note that the stationary distribution appears in each row.

Detailed balance

Computation of the stationary distribution for a large chain can be time consuming, so it is worthwhile to have sufficient criteria for stationarity.

Theorem: Suppose that η is a probability distribution on the state space of an ergodic chain satisfying the *detailed balance* condition

$$\forall xy[\eta_x p_{x.y} = \eta_y p_{y,x}].$$

Then η equals π, the stationary distribution.

Proof: Since each state must have some predecessor, we have for any x, y

$$\sum_x \eta_x p_{xy} = \sum_x \eta_y p_{yx} = \eta_y \sum_x p_{yx} = \eta_y.$$

We can give it the following interpretation. Since we expect to be in the state x a fraction π_x of the time, detailed balance asserts that the average traffic across the edge $x \to y$ is the same as the average traffic across $y \to x$.

Our example satisfies the detailed balance condition. This always holds when $x = y$, so we only have to consider $x = A$, $y = B$. Then

$$\pi_A p_{A,B} = 1/6 \times 1/2 = 5/6 \times 1/10 = \pi_B p_{B,A}.$$

Not every Markov chain with a stationary distribution satisfies detailed balance. Consider, for example, the chain for which

$$\begin{aligned} \Pr[A \to B] &= 1/2, \\ \Pr[A \to C] &= 1/2, \\ \Pr[B \to C] &= 1, \\ \Pr[C \to A] &= 1. \end{aligned}$$

All traffic goes forward along the cycle A, B, C, A.

Chains satisfying detailed balance are called *reversible*.

Notes

The Metropolis algorithm for cooking up a chain with a desired invariant distribution needs to be mentioned. There is a nice cryptanalysis example in P. Diaconis, The Markov Chain Monte Carlo Revolution, Bull. AMS, 2009.

Card shuffling is a great example, with connections to the previous "exact analysis" topics we have studied. See N. Berestycki, Notes on Card Shuffling, manuscript, n.d.

Lecture 41

Topic du jour: MCMC for graph coloring

M. Jerrum, Mathematical Foundations of the Markov Chain Monte Carlo Method, in Habib et al, pp. 116-165.

Background from last lecture

Finite Markov chains: finite state transition diagrams with probabilities p_{xy} on the edges.

Ergodic chains: strongly connected, no periodicity. This implies that there is a stationary distribution to which probabilities converge.

Detailed balance ($\pi_x p_{xy} = \pi_y p_{yx}$ is a sufficient condition for π to be the stationary distribution.

Graph coloring

Let $G = (V, E)$ be an undirected graph. The degree of a vertex v is

$$d(v) = \#\text{ of neighbors of } v\ .$$

We also let

$$\Delta = \max\{d(v) : v \in V\}.$$

Let Q be a finite set of colors. We write q for the number of colors, so that $q = |Q|$. A *coloring* is a mapping $c : V \to Q$ that that assigns different colors to the ends of each edge.

If Q is large enough, there will always be a coloring. For example, it suffices that $q = |V|$. More to the point is the following result.

Theorem: If $q \geq \Delta + 1$, then G has a q-coloring.

To prove this we use a simple greedy algorithm. Number the vertices $1, \ldots, n$. For $i = 1, \ldots, n$ we assign i any color not already assigned to one of its neighbors.

How hard is it to tell if a graph has a q-coloring? This question is

Trivial if $q = 1$. (The graph can have no edges.)

Answerable in linear time if $q = 2$. (Use, for example, depth-first search to find a coloring. If this fails you have found an odd cycle and a 2-coloring is impossible.)

NP-complete if $q = 3$.

We will now address the following question: How can we pick a random coloring of G? This is not, in general, a trivial task since there can be a large number of them.

A Markov chain for graph colorings

We now define a Markov chain on Ω, the set of q-colorings of G. To go from one coloring to the next, we execute the following step:

> Choose a vertex $v \in V$ uniformly. Assign v a color chosen uniformly, subject only to the rule that it differ from all colors assigned to v's neighbors.

Note that it is possible for a step in the chain to keep the same coloring.

Theorem: If $q \geq \Delta + 2$, this chain is ergodic.

To prove this we must show 1) strong connectivity 2) the cycle lengths have gcd equal to 1.

> To prove 1), suppose $c_1, \ldots, c_n$ is a coloring of the n vertices. We can change any other coloring to this one by a sequence of one-step moves, given by the following procedure. The idea is to successively fix the colors of vertex 1, then vertex 2, and so on.
>
> ```
> for i = 1, 2, ..., n:
>     for j > i and adjacent to i:
>         if j is colored c_i, change its color
>             to be different from c_i or the
>             color assigned to any neighbor of j
>         assign i the color c_i.
> ```
>
> We need to think a bit about why this works. First, since $c_1, \ldots, c_n$ is a coloring, every j with $j < i$ has already been assigned the color $c_j \neq c_i$. Since $q > \Delta + 1$, there is a sufficient supply of colors to paint the neighbors with $j > i$ as required by the algorithm.
>
> To prove 2) it suffices to observe that the algorithm is allowed to keep the same coloring. So there are cycles of length 1, making the gcd equal 1.

What is the stationary distribution?

> We will use detailed balance to show that the stationary distribution is uniform.
>
> Suppose c' is a neighbor of c in Ω. Then these colorings are the same except for one vertex v. Let there be δ colors among the neighbors of v. Then we have
>
> $$\Pr[c \to c'] = \frac{1}{n} \cdot \frac{1}{q - \delta} = \Pr[c' \to c].$$
>
> Multiplying both sides by $1/|\Omega|$, we have detailed balance.
>
> It should be noted that we do not have to know $|\Omega|$ for this calculation to proceed.

Sampling a nearly random coloring

> To find a coloring nearly uniformly distributed among all the colorings of G, we can do the following. First, start with any coloring, for example the greedy coloring that is guaranteed by $q \geq \Delta + 1$. Second, run the chain for a long time and stop. The theory of Markov chains tells us that the coloring will be nearly uniformly distributed.

In the next lecture we will treat this problem quantitatively, but it should be clear already that this will work well when $|Q| \gg \Delta$. To see this we use the coupon collector's problem. The idea is that uniformity is achieved, or nearly so, when each vertex has been repainted at least once. The expected time at which this happens is $\sim n \log n$.

Lecture 42

Topic du jour: Analysis of MCMC algorithm to select a random coloring of a graph.

M. Jerrum, Mathematical Foundations of the Markov Chain Monte Carlo Method, in Habib et al, pp. 116-165.

What we've done so far

Last time we described a Markov chain that walks among the q-colorings of a graph. The idea is to repeatedly select a random vertex, and give it a random color different that the colors of its neighbors.

We proved that this chain is ergodic if $q \geq \Delta + 2$, where Δ is the degree (largest number of neighbors of any vertex) of the graph.

Using detailed balance, we showed that the stationary distribution is uniform.

Our goal is now to show that when q is large enough, the chain quickly gets close to the stationary distribution.

Total variation distance.

Suppose that π and η are two probability distributions on a finite set Ω. Their *total variation distance* is

$$d(\pi, \eta) = \max_{A \subset \Omega} |\pi(A) - \eta(A)|.$$

As defined this is a little hard to compute, since one must potentially consider $2^{|\Omega|}$ sets. As an exercise you can prove that

$$d(\pi, \eta) = \frac{1}{2} ||\pi - \eta||_1,$$

where

$$||\pi - \eta||_1 = \sum_x |\pi_x - \eta_x|$$

is the L_1 norm.

Because of this, d satisfies the usual axioms of a distance function:

$$d(\pi, \eta) = 0 \text{ iff } \pi = \eta,$$

$$d(\pi, \eta) = d(\eta, \pi),$$

and

$$d(\pi, \rho) \leq d(\pi, \eta) + d(\eta, \rho).$$

Mixing time

Recall that if π is the stationary distribution for a finite Markov chain, and π_t is the distribution of X_t, then

$$\lim_{t\to\infty} \pi_t = \pi.$$

This is equivalant to saying that

$$\lim_{t\to\infty} d(\pi_t, \pi) = 0.$$

We now introduce a way to measure how fast this convergence is. For $X_0 = x$ (that is, we start the system in state x), let

$$\tau_x(\epsilon) = \min\{t : \forall t' \geq t[d(\pi_{t'}, \pi) \leq \epsilon]\}$$

This definition looks complicated but the idea is not. Since the distance converges to 0 there will be a first time at which the distance dips below ϵ and stays there forever. This time is $\tau_x(\epsilon)$.

The *mixing time* of the chain is

$$\tau(\epsilon) = \min\{\tau_x(\epsilon)\}.$$

As an exercise, you should prove that for any initial state, if $t \geq \tau(\epsilon)$,

$$d(\pi_t, \pi) \leq \epsilon.$$

(Condition on the initial state.)

Our intuition is that τ will get larger and larger as $\epsilon \to 0$.

Coupling

The following device has proved to be very useful in understanding the behavior of Markov chains.

Definition: A pair of processes (X_i, Y_i), $i = 0, 1, 2, \ldots$ is called a *coupling* if each component, considered in isolation, is a sample from the Markov chain.

We state the condition formally as

$$\Pr[X_{i+1} = z | X_i = x, Y_i = y] = p_{x,z}$$

and

$$\Pr[Y_{i+1} = w | X_i = x, Y_i = y] = p_{y,w}$$

This is usually used in the following way. Think of $X_0, X_1, \ldots$ as the process we want to study. We start Y_i in the stationary distribution, and design the coupling so that at some random time t,

$$X_t = Y_t.$$

We know that Y_t has the stationary distribution, and therefore X_t does too. We can if we want switch X_t so that it follows Y_t after this point.

Let T be the first time t at which $X_t = Y_t$. We will call this the *collision time*.

Our definition has the following implication. If we run X and Y until the collision time, and output whatever state X happens to be in, that state has the stationary distribution. But we are *not* allowed to conclude the stronger statement

$$\Pr[X_T = x | T = k] = \pi_x.$$

[Need to explain why.]

Our next theorem says that a tail bound on the collision time gives us a bound on the mixing time. For this result we will assume that X and Y are "coalesced" after they collide. That is, if $X_t = Y_t$ for some t, then $X_{t'} = Y_{t'}$ for all $t' \geq t$.

Theorem: Let t (thought of as a function of ϵ) be such that

$$\Pr[X_t \neq Y_t] \leq \epsilon.$$

Then t is an upper bound on the mixing time.

Proof: Let $u \geq t$. Then for any $A \subset \Omega$,

$$\begin{aligned}
\Pr[X_u \in A] &\geq \Pr[Y_u \in A \ \& \ X_u = Y_u] \\
&= 1 - \Pr[Y_u \notin A \vee X_u \neq Y_u] \\
&\geq 1 - \Pr[Y_u \notin A] - \Pr[X_u \neq Y_u] \\
&\geq \Pr[Y_u \in A] - \epsilon \\
&= \pi(A) - \epsilon,
\end{aligned}$$

since $u \geq t$. From the same result for $\bar{A}$, we infer

$$\Pr[X_u \in A] \leq \pi(A) + \epsilon.$$

Thus $d(\pi_u, \pi) \leq \epsilon$ for all $u \geq t$, which shows the mixing time is $\leq t$.

A lemma about distributions with given marginals

Lemma: Let A, B be two subsets of a finite set Ω. There are two random variables x and y such that: 1) x has all its mass on A, where it is uniformly distributed; 2) the same for y and B; 3) we have

$$\Pr[x = y] = \frac{|A \cap B|}{\max\{|A|, |B|\}}.$$

Proof: Let's handle the degenerate cases first. If $A = B$, then we use one sample from A for both. If $A \cap B = \emptyset$, then choose x and y independently. Otherwise, without loss of generality assume that

$$0 < |A \cap B| < |A| \leq |B|.$$

We now choose $x \in A$. If it lands in $A \cap B$ we use this for y as well. If not, we sample from a distribution on B that has mass

$$\frac{|B|^{-1} - |A|^{-1}}{\Pr[x \in A \cap B]}$$

on each element of $A \cap B$ and divides its remaining mass uniformly on $B - A$.

Evidently this satisfies 1) and 3). To prove 2) we observe that if $y \in I = A \cap B$ we have

$$\begin{aligned}\Pr[y] &= \Pr[y \ \& \ x \in I] + \Pr[y \ \& \ x \notin I] \\ &= 1/|A| + \Pr[y | x \notin I]\Pr[x \notin I] \\ &= 1/|B|.\end{aligned}$$

By symmetry, the probabilities of $y \notin I$ are all equal, so they are $1/|B|$ too.

Jerrum calls this an "easy exercise" so there is probably a more elegant proof.

Rapid mixing for the graph coloring MCMC

We will assume that $q > 2\Delta$. That is, we have roughly twice the number of colors as are needed by the greedy coloring procedure.

Now we define a coupling. The X chain will start with a particular coloring c. The Y chain will start in the uniform distribution. A step in the evolution of these processes does the following:

Choose a vertex v uniformly. Let A be the set of feasible colors for v under X, and B the set of feasible colors for v under Y. (That is, we can use a color unless it is already on one of v's neighbors.) Sample new colors from the distribution whose existence is guaranteed by the lemma.

Before doing any analysis we should think a little about how this will work. If there are many colors the probability of agreement is close to 1. Thus, we expect to see roughly the following kind of behavior. Vertices are selected at random and given (frequently) the same color by both X and Y. After not too many steps each vertex will have been chosen, and if all colors are the same, the process stops.

To analyze this we consider D_t, the set of vertices at which the two colorings disagree at time t. Let us write N_t for $|D_t|$. Since we repaint only one vertex at at time,

$$-1 \leq N_{t+1} - N_t \leq 1$$

So N_t executes a complicated kind of random walk on $\{0, \ldots, n\}$. Note that 0 is an absorbing point, once the two colorings are equal they stay equal forever.

Our goal is now to prove that for some $a > 0$,

$$E[N_{t+1}] \leq (1 - a)E[N_t].$$

We need some notation first. Call an edge an "agree-disagree" edge if the colorings are the same on one of its vertices and different on the other. We will write $d'(v)$ for the number of agree-disagree edges incident on v. Note that

$$\sum_{v \in D_t} d'(v) = \sum_{v \notin D_t} d'(v). \tag{1}$$

Now let A_v be the possible colors that the X chain can assign v, B_v the same for Y, and $I = A_v \cap B_v$.

If the colors agree on v, we have

$$|A_v| \leq |I| + d'(v),$$

since $A_v \subset I \cup (A_v \cap \bar{I})$. The same holds for B_v, so

$$\max\{|A_v|, |B_v|\} \leq |I| + d'(v).$$

We also have $|\bar{I}| \leq \Delta + d'(v)$ (each neighbor of v knocks out one or two colors, depending on whether these agree or disagree), so

$$|I| + d'(v) \geq q - \Delta.$$

This gives

$$\begin{aligned} \Pr[\text{ new colors disagree } |v] &= 1 - \frac{|I|}{\max\{|A_v|, |B_v|\}} \\ &\leq 1 - \frac{|I|}{|I| + d'(v)} \\ &= \frac{d'(v)}{|I| + d'(v)} \\ &\leq \frac{d'(v)}{q - \Delta}. \end{aligned} \tag{2}$$

We do a similar calculation assuming the colors disagree on v. If a color can be used by X, it can either be used by Y as well, or only Y assigns it to a neigbor of v. There are at most $\Delta - d'(v)$ of the latter, so

$$|A_v| \leq |I| + \Delta - d'(v).$$

By symmetry we conclude

$$\max\{|A_v|, |B_v|\} \leq |I| + \Delta - d'(v).$$

We also have $|\bar{I}| \leq 2\Delta - d'(v)$, so

$$|I| \geq q - 2\Delta + d'(v).$$

This gives

$$\begin{aligned}\Pr[\text{ new colors agree } |v] &= \frac{|I|}{\max\{|A_v|, |B_v|\}} \\ &\geq \frac{|I|}{|I| + \Delta - d'(v)} \\ &\geq \frac{q - 2\Delta}{q - \Delta} + \frac{d'(v)}{q - \Delta}\end{aligned} \tag{3}$$

We now condition on the vertex selected:

$$E\left[N_{t+1} - N_t\right] = \sum_v E[N_{t+1} - N_t | v] \Pr[v]$$

Grouping the vertices according to whether their colorings agree or disagree and using (1)–(3), we find that

$$E[N_{t+1} - N_t | N_t] \leq \frac{1}{n} \sum_{v \notin D_t} \left(\frac{d'(v)}{q - \Delta} \right) - \frac{1}{n} \sum_{v \in D_t} \left(\frac{q - 2\Delta}{q - \Delta} + \frac{d'(v)}{q - \Delta} \right) = -aN_t,$$

with

$$a = \left(\frac{q - 2\Delta}{q - \Delta} \right) \cdot \frac{1}{n}.$$

This gives the result we want: $E[N_{t+1}] \leq (1 - a)E[N_t]$.

We can now bound the mixing time.

By induction on t,

$$E[N_t] \leq (1 - a)^t E[N_0] \leq e^{-at} n.$$

So

$$\Pr[N_t \neq 0] \leq \epsilon$$

provided that

$$e^{-at} n < \epsilon,$$

that is, provided that

$$t > a^{-1} \log(n/\epsilon).$$

We observe that $a^{-1} \leq (\Delta + 1)n$, which estimates the mixing time as

$$\tau(\epsilon) \leq (\Delta + 1) n \log(n/\epsilon).$$

Notes

This should be replaced by an argument involving path coupling. See Dana Randall's survey talk from FOCS 2003.

CS 709
Eric Bach
Spring 2001

Lecture 43

Topic du jour: Branching processes

L. Devroye, Branching Processes and Their Applications in the Analysis of Tree Structures and Tree Algorithms, in Habib et al, pp. 249-314; T. Harris, *Branching Processes*.

Galton-Watson Processes

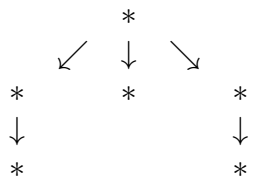
This is a tree-growing stochastic process.

We start with a root. Give it a random number of children. Do the same for each of the children, their children, and so on.

Each individual draws the number of its children from the same distribution, independently.

The resulting tree can be finite or infinite.

Here's an example:



The generations have sizes 1,3,2.

Clearly, the more children a node can have, the larger the tree will be, on average. To analyze this we introduce the random variable

$$Z = \ \#\text{ of children of a node .}$$

The key parameter in analyzing the behavior of the tree is the so-called Malthusian parameter

$$m = E[Z].$$

(We'll assume this is finite.)

Let Z_i be the number of individuals in the i-th generation, so that $Z_0 = 1$ (the root). The averages of these numbers are very well behaved.

Theorem: We have $E[Z_n] = m^n$.

We can prove this by induction. The base case is the root. To go from $n-1$ to n, take expectations of

$$E[Z_n | Z_{n-1}] = mZ_{n-1}$$

to get

$$E[Z_n] = mE[Z_{n-1}] = m \cdot m^{n-1} = m^n.$$

It follows from this that $W_n = Z_n/m^n$ forms a martingale. Doob's limit law for martingales (which we won't prove) implies that W_n converges to a random variable W.

Let $f_n(s) = \sum_{k \geq 0} \Pr[Z_n = k] s^k$. This is the probability generating function (PGF) for Z_n, and it equals $E[s^{Z_n}]$. Then f_n satisfies a (generally nonlinear) recurrence relation.

Theorem: We have $f_0 = s$, and for $n \geq 1$, $f_n(s) = f(f_{n-1}(s))$.

Again, this can proved by induction. Since PGF's multiply when you take a sum of independent random variables, we have

$$E[s^{Z_n} | Z_{n-1}] = f(s)^{Z_{n-1}} = f_{n-1}(s).$$

Upon taking expectations,

$$E[s^{Z_n}] = E[f(s)^{Z_{n-1}}] = f_{n-1}(f(s)) = f(f_{n-1}(s)).$$

Some examples

Let us take

$$Z = \begin{cases} 0, & \text{with probability } p_0 \ ; \\ 1, & \text{with probability } p_1 \ . \end{cases}$$

We know the behavior of this process. The root makes a linear chain with a geometrically distributed number of nodes.

Here it is explicitly possible to compute the generating function, since

$$f(s) = p_0 + p_1 s.$$

Applying our machinery for solving linear recurrence relations, we get

$$\begin{aligned} f_n(s) &= p_0(1 + p_1 + \ldots + p_1^{n-1}) + p_1^n s \\ &= (1 - p_1^n) + p_1^n s. \end{aligned}$$

So unless $p_1 = 1$, we have $\Pr[Z_n = 0] = 1 - p_1^n \to 1$.

Let I_A denote the indicator function for an event A. Evidently we have

$$I_{Z_1=0} \leq I_{Z_2=0} \leq I_{Z_3=0} \leq \cdots \to I_{\exists n Z_n = 0}.$$

So by monotone convergence

$$\Pr[\exists n Z_n = 0] = \lim_{n \to \infty} \Pr[Z_n = 0] = 1.$$

This means that extinction is certain.

Here is another example where it is possible to see explicitly what is going on. Let

$$\Pr[Z = k] = (1/2)^{k+1}.$$

That is, to determine the number of children, we flip a fair coin and count the number of tails before the first head.

The PGF for this distribution is a linear fractional transformation:

$$f(s) = \frac{1/2}{1 - s/2} = \frac{1}{-s + 2}.$$

Thus

$$f_n(s) = \frac{a_n s + b_n}{c_n s + d_n},$$

where

$$\begin{pmatrix} a_n & b_n \\ c_n & d_n \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -1 & 2 \end{pmatrix}^n.$$

It is not hard to show by induction that

$$\begin{pmatrix} a_n & b_n \\ c_n & d_n \end{pmatrix} = \begin{pmatrix} -n+1 & n \\ -n & n+1 \end{pmatrix}$$

So the PGF for the size of the n-th generation is

$$f_n(s) = \frac{1 - (1 - n^{-1})s}{(1 + n^{-1}) - s},$$

which converges as $n \to \infty$ to 1. So

$$\Pr[Z_n = 0] \to 1.$$

As before we can use this to show that the resulting tree is finite with probability 1.

When is a Galton-Watson tree finite?

There are three cases, depending on how m, the mean number of children, compares to 1. Proofs of these results can be found in Harris.

Subcritical.

Here we have $m < 1$. In this case, the tree is finite with probability 1.

Critical.

Here we have $m = 1$. In this case also, the tree is finite with probability 1, with one exception.

The exception is the degenerate case where Z is always 1.

Supercritical.

Here we have $m > 1$. Then

$$q := \Pr[\text{ the tree is finite }] < 1.$$

Generally, q is positive. The only exception is if the distribution of Z has no mass at 0. (In all other cases, there is a positive probability that the root will have no children.)

Remarkably, a supercritical Galton-Watson tree conditioned on extinction is another Galton-Watson tree with the "discounted" generation size Z^*. We have

$$\Pr[Z^* = k] = q^{k-1}\Pr[Z = k].$$

The new mean number of children is

$$m^* = E[Z^*] < 1.$$

(Why is it less than 1?)

This is not hard to prove. Since the children reproduce independently, the probability that the root has k children, each of which has a finite subtree, is $\Pr[Z = k]q^k$. Dividing this by q gives the above formula for

$$\Pr[Z = k|\text{ finite tree }] = \Pr[Z^* = k].$$

It should not be missed that this gives us a way to compute q. Since we have a probability distribution, it must be the case that $q = f(q)$, where f is the probability generating function for Z. (See Devroye, pp. 250-251 on this point.)

Notes.

In future editions of this course it would be well to do this material along with the other random tree models.

An obvious question to ask is, how do the various models relate? For example, can random binary trees be obtained from Galton-Watson trees by conditioning? Do these two give "large" trees that are roughly equivalent? Section 6 of Devroye's paper has some information about this.

Lecture 44

Topic du jour: Searching trees with random costs.

L. Devroye, loc. cit.; R. Karp and J. Pearl, Artificial Intelligence, v. 21, 1983, 99-117.

Review of branching processes

We grow random trees in the following manner. Start with a root, which makes a random number of children. Each child then reproduces itself in the same manner.

The Galton-Watson process is one in which the reproduction distribution is the same for all nodes.

Everything about the process can be derived if you know the PGF of Z, the number of children for a given node. It is

$$f(s) = E[s^Z].$$

In particular, $m := E[Z] = f'(1)$ controls the eventual fate of the tree.

If $m \leq 1$ the tree is almost surely finite, except for one degenerate case.

If $m > 1$ the tree is finite with probability $q < 1$. There is no formula for q but it can be shown that q is the unique solution in $[0, 1)$ to the equation

$$f(q) = q.$$

We can compute q by iteration. Whenever $0 \leq q_0 < 1$,

$$q = \lim_{n \to \infty} f^{(n)}(q_0).$$

The mean size of the n-th generation (counting the root as generation 0) is m^n.

Searching Galton-Watson Trees

In many situations we want to quickly find a "deep" node in a random tree. Depth-first search is the obvious strategy for this purpose.

Theorem: Let T be tree built by a Galton-Watson process for which Z is bounded. (That is, there is some B such that $\Pr[Z > B] = 0$.) Let D_n be the number of nodes visited by a depth-first search that stops as soon as it reaches a node of level n. Then $E[D_n] = O(n)$.

Proof: We have various cases depending on $m = E[Z]$.

If $m \leq 1$, then the expected size of the tree down to and including level n is

$$1 + m + m^2 + \ldots + m^n \leq n + 1.$$

So we now assume the process is supercritical, that is, $m > 1$.

Since

$$E[D_n] = E[D_n||T| < \infty] \cdot q + E[D_n||T| = \infty] \cdot (q - 1)$$

it suffices to bound the two conditional probabilities. Given $|T| < \infty$, the tree is subcritical, and we are reduced to the previous case.

Given $|T| = \infty$, a subtree can either be finite (we say the node is mortal) or infinite (immortal). The expected size of a finite tree is

$$1 + m^* + {m^*}^2 + \ldots = \frac{1}{1 - m^*} = M < \infty.$$

(Here m^* is the expected number of children given that the tree is finite. It can be shown that this is $f'(q)$.) At each level, the search will go through at most B mortal nodes, below which we expect to find $\leq MB$ nodes. Then it goes through an immortal node, never to return. This gives a total expected cost of

$$(MB + 1)n = O(n).$$

As an exercise, you should verify that this theorem does not hold for breadth-first search, unless $m \geq 1$.

Karp-Pearl Search Model

With the goal of understanding the behavior of heuristic search processes, Karp and Pearl introduced the following model.

Take an infinite binary tree in which each node has exactly two children. Give each edge a random cost, 0 or 1, independently. (We will assume that the edge costs are identically distributed.)

For any node, there is a unique path to the root. We assign it a cost which is the sum of the edge costs along this path.

We are interested in questions like the following. How are the costs of individual nodes distributed? How can I find a "deep" node with minimal cost, or cost not too far from the minimum?

The key parameter for such questions is

$$p = \Pr[\text{ edge cost 1 }].$$

What is the cost of a level n node?

Let x be a particular node at level n and C_n its cost. Since the path to the root has length n, C_n has a Binomial(n, p) distribution.

Search by cost order

To find a least-cost node at level n, the following strategy seems reasonable:

Search out of the root and identify all cost 0 nodes. If a depth n node is encountered, we can stop the search since its cost is obviously minimal. If not, the search has been blocked by various cost-1 edges. Search out of the ends of these to identify all cost 1 nodes, then repeat to get cost 2 nodes, and so on.

A hydraulic metaphor is appropriate for this. Think of a 0 edge as an open pipe, and a 1 edge as closed. The search can be thought of as a process that successively opens a blocked pipe and floods a new set of dry nodes.

If $p < 1/2$ (so we expect that most edges will have cost 0), this strategy takes linear time.

To prove this, we observe that the cost 0 edges out of a particular node form a Galton-Watson tree. The expected number of children for this tree is $2(1-p) > 1$ so this process is supercritical. We can think of the search through each cost 1 edge as a coin flip with success probability $1-q$. (We are sucessful when the tree is infinite, which means that we don't have to consider any more cost 1 edges.) The expected number of coin flips for a success is $1/(1-q)$, and each trial (successful or not) costs $O(n)$, so the total is $O(n)$.

To get some practice with the Galton-Watson model the following looks like a good exercise. Determine the bound explicitly (it should depend on p).

What happens for other values of p?

If $p = 1/2$ the cost is $O(n^2)$.

For $p > 1/2$ the cost is exponential.

Notes.

The implied constant in the result for depth-first search depends on the distribution. I am not sure if this is necessary.

Lecture 45

Topic du jour: Branching random walks.

L. Devroye, loc. cit.; R. Karp and J. Pearl, Artificial Intelligence, v. 21, 1983, 99-117.

What are branching random walks?

In the last two lectures, we have studied two kinds of "tree processes." We considered trees with a random shape (Galton-Watson trees), and then considered fixed shape trees with random edge costs.

The branching random walk is a model that combines features of both.

Definition: A *branching random walk* is defined by a Galton-Watson process (which makes a random tree), plus a probability distribution that is used to select a number for each edge. These numbers are independent and identically distributed.

As in the last lecture, the cost of a node is the sum of the edge numbers on its path to the root.

The Karp-Pearl search model is an example of this. Here the Galton-Watson process is deterministic (each node has two children, never more, never less). The edge numbers are selected by Bernoulli trials.

Two random variables of interest are

$$B_n = \min\{ \text{ cost of a level } n \text{ node } \}$$

and

$$D_n = \max\{ \text{ cost of a level } n \text{ node } \}$$

If the edge cost distribution is bounded, say $0 \leq \text{ cost } \leq 1$, then there is a duality between these. Let B'_n and D'_n be the corresponding random variables for the costs $C' = 1 - C$. Then

$$B_n = n - D'_n$$

and

$$D_n = n - B'_n.$$

Variants of the basic model

The following "lifetime" picture is often used. Start with one individual, who lives for a random time, and then reproduces to form a random number of individuals with identical behavior. This is the same as our model, except that the degree of the root is always 1. Analogs of B_n and D_n in this model are times of the first birth and last death in the n-th generation.

Instead of lifetimes, one can think of displacements in space. This is a convenient metaphor when the edge costs could be negative. Start with one individual at

location 0. At times $t = 0, 1, 2, \ldots$ each individual makes a random number Z of copies of itself, each of which moves independently to a new location. If $Z \equiv 1$ this reduces to an ordinary random walk, hence the name "branching random walk." The successive generations spread out in space, and the roles of B_n and D_n are played by the leftmost and rightmost members of generation n.

The BHK Theorem

In a branching random walk the minimum and maximum costs at level n (B_n and D_n) are sharply predictable. Around 1975, Biggins, Hammersley, and Kingman proved (under mild restrictions on distributions involved) that there are constants α and β for which

$$B_n \sim \beta n,$$
$$D_n \sim \alpha n$$

with probability 1.

Their recipe for computing α and β uses the moment generating function, which we now introduce.

Let X be a random variable for the displacement distribution. Its *moment generating function* is

$$\phi(\theta) = E[e^{\theta X}].$$

Note that this is the exponential (!) generating function for the moments of X. If $X \geq 0$ and is integer-valued, ϕ is its PGF evaluated at e^θ.

Most of the time we use the MGF as a formal series. Here, we are interested in it as a function of the real variable θ.

Let m be the mean number of children for the underlying branching process. We define

$$\psi(\theta, a) = me^{-\theta a}\phi(\theta), \qquad (= mE[e^{\theta(X-a)}])$$

$$\Phi_+(a) = \inf\{\psi(\theta, a) : \theta > 0\},$$

and

$$\Phi_-(a) = \inf\{\psi(\theta, a) : \theta < 0\},$$

Theorem: Let the edge costs be nonnegative, with a moment generating function that is analytic around 0. Then

$$\alpha = \inf\{a > 0 : \Phi_+(a) < 1\}$$

and

$$\beta = \inf\{a > 0 : \Phi_-(a) > 1\}.$$

We won't prove this but a large deviation argument makes the recipe for α plausible.

Let Y be the sum of n i.i.d. copies of X. By the same argument we used for the Chernoff bound,

$$\Pr[Y \geq an] \leq \left(e^{-\theta a}\phi(\theta)\right)^n.$$

Given that there are k level n nodes, by the union bound,

$$\Pr[\ (\exists \text{ node of cost} \geq an) \mid k \text{ level } n \text{ nodes }] \leq k\left(e^{-\theta a}\phi(\theta)\right)^n.$$

Now take expectations of this and note that $E[k] = m^n$ to conclude

$$\Pr[\ (\exists \text{ node of cost} \geq an)] \leq \psi(\theta, a)^n.$$

The recipe chooses α to be the lower limit of the a's for which this bound decreases exponentially.

A similar case can be made for β.

Application to Karp-Pearl trees

The edge cost X equals 1 with probability p and 0 with probability q. Also $m = 2$. So

$$\psi(\theta, a) = 2[pe^{(1-a)\theta} + qe^{-a\theta}] \geq 0.$$

We have

$$\frac{\partial \psi}{\partial \theta} = 2e^{-a\theta}\left[p(1-a)e^{\theta} - qa\right].$$

If $a \geq 1$, this is ≤ 0, and ψ decreases exponentially to 0 on $[0, \infty)$. Now let $0 < a < 1$. Under these conditions, $\psi \to +\infty$ as $\theta \to \pm\infty$, so the minimum of ψ occurs when $\frac{\partial \psi}{\partial \theta} = 0$, i.e. when

$$e^{\theta} = \frac{aq}{(1-a)p}.$$

Furthermore, the minimizing θ has the sign of $a - p$.

So for $0 < a < 1$ we define

$$M(a) := \min_{\theta} \psi(\theta, a) = 2\left[p\left(\frac{aq}{(1-a)p}\right)^{1-a} + q\left(\frac{(1-a)p}{aq}\right)^{a}\right].$$

Note that this is symmetric with respect to

$$\left\{\begin{matrix} a \\ p \end{matrix}\right\} \leftrightarrow \left\{\begin{matrix} 1-a \\ q \end{matrix}\right\}.$$

We now think of α and β as functions of p. By our duality result,

$$\beta(p) = 1 - \alpha(1-p).$$

So we only have to compute one of these. Let's determine α.

By our analysis of $\partial\psi/\partial\theta$, we have

$$\Phi_+(a) = \begin{cases} 2 \quad (= \psi(0,a)), & \text{if } 0 < a \le p; \\ M(a), & \text{if } p < a < 1; \\ 0, & \text{if } 1 < a. \end{cases}$$

This is continuous at $p = a$. With some work (factor out $(...)^a$, simplify, take logs, differentiate) it is possible to show that $M(a)$ decreases. Furthermore, $\lim_{a\to 1} M(a) = 2p$.

So we have

$$\alpha = \begin{cases} \text{solution to } M(a) = 1, & \text{if } p < 1/2; \\ 1, & \text{if } p \ge 1/2. \end{cases}$$

The equation for α can be solved numerically. Here are some values.

p	α	β
0.0	0.000000	0.000000
0.1	0.577490	0.000000
0.2	0.747020	0.000000
0.3	0.864757	0.000000
0.4	0.948695	0.000000
0.5	1.000000	0.000000
0.6	1.000000	0.051305
0.7	1.000000	0.135243
0.8	1.000000	0.252980
0.9	1.000000	0.422510
1.0	1.000000	1.000000

It is interesting to note that the effect of probabilities close to 0 or 1 is rather more than one might suspect.

There is a nice graph of β on p. 108 of Karp and Pearl.

What does BHK tell us about searching for low-cost nodes?

The interesting case here is $p > 1/2$. (Karp and Pearl give an $O(n)$ algorithm for $p < 1/2$, and an $O(n^2)$ algorithm for $p = 1/2$.)

The cost of the minimum path is very likely to be close to β (a constant that we can compute) times the depth to which we wish to explore.

Colin McDiarmid and Gregory Provan (IJCAI 1991) discuss the following bounded lookahead algorithm. (They call this A3.)

> Choose a fixed level L. Search all paths down to level L, and then choose one of minimum length. Search in the same manner out of the end of this path, stopping when level n is reached.

For convenience we assume n to be a multiple of L.

This procedure examines

$$(2^{L+1}-1)\frac{n}{L} = O_L(n)$$

nodes.

Analysis of bounded lookahead:

Let C_n be the cost of the path found by the algorithm

Let B_n be the cost of the optimal path

Theorem: For any $\epsilon > 0$ we can choose L so that (with probability 1)

$$C_n \leq (1+\epsilon)B_n.$$

In other words, the approximation factor for bounded lookahead can be made arbitrarily close to 1.

Proof: Use BHK. Suppose $\delta > 0$ (we will pick it later). Choose L so that $(1/L)E[B_L] \leq (1+\delta)\beta$. (A fine point: the BHK theorem only gives us almost sure convergence. But B_n/n is bounded, so the first moments converge as well.) The length of the path found by the algorithm, C_n is the sum of n/L independent random variables distributed like B_L. Using the usual strong law of large numbers,

$$\frac{C_n}{n/L} \sim E[B_L],$$

and (divide through by L),

$$\frac{C_n}{n} \sim \frac{1}{L}E[B_L] \leq (1+\delta)\beta.$$

Using BHK again, for sufficiently large n we have (almost surely)

$$\frac{B_n}{n} \geq (1-\delta)\beta.$$

Combining these, we now choose δ so that

$$\frac{C_n}{B_n} \leq \frac{1+\delta}{1-\delta} \leq (1+\epsilon).$$

Notes

Karp and Pearl analyzed other heuristics related to A3. See also two papers by Colin McDiarmid (referenced in Devroye's article).

Karp and Pearl also show that any process that is guaranteed to determine an optimal cost node at level n must take exponential (expected?) time.

For the BHK theorem, see (in historical order)

J. M. Hammersley, Postulates for subadditive processes, Ann. Probab. 2 (1974), 652-680;

J. F. C. Kingman, The first birth problem for an age-dependent branching process, Ann. Probab. 3 (1975), pp. 790-801;

J. D. Biggins, The first and last birth problems for a multitype age-dependent branching process, Adv. Appl. Probab. 8 (1976), pp. 446-459.

Lecture 46

Topic du jour: Introduction to lower bounds.

Why lower bounds?

One of the goals of theoretical computer science is to determine the absolute limits on our ability to solve problems using computation. This is similar in spirit to physical theories such as thermodynamics, in which it is proved that the efficiency of a heat engine is strictly limited by a numerical factor depending on the engine's operating temperatures.

To date this program has not met with unqualified success. However, there are some areas in which results are known.

1. Recursive function theory: Certain problems such as the halting problem for Turing machines cannot be solved by algorithms.
2. It can be shown that there are arbitrarily complex computational problems. In any "reasonable" model of computation, there are problems whose solution requires more steps, as a function of the input size, than any given function f.
3. The theory of NP-completeness provides a long list of likely candidates for natural problems having no polynomial time algorithm.
4. If we restrict attention to algorithms using a limited set of plausible operations, it is often possible to show that at least a certain number of these operations must be used. An example is the result that to sort by key comparisons requires $\geq \log_2 n!$ work.
5. In certain cases, we know exponential lower bounds for severely limited models of computation. For example, constant-depth circuit families with unbounded fan-in cannot compute the parity function and remain polynomial size.

We'll concentrate on 4, an area often called *structured lower bounds* or *concrete complexity*. Results in this area come in two flavors:

Combinatorial (bounds on comparisons, messages sent, memory references, etc.).

Algebraic (bounds on arithmetic operations, usually multiplications).

As a general rule, lower bounds on computational complexity come from observing some non-trivial global property of a set associated with the solution to a problem. Examples:

Topological invariants (number of connected components, etc.) of geometric sets.

Entropy of a probability distribution.

Degree of a mapping or set of algebraic equations.

As you can see, this will require us to become familiar with a wide range of mathematical ideas.

Another approach is akin to the model building done by mathematical logicians. The idea is to let the algorithm itself construct a bad input (one requiring lots of time).

This is often called the adversary method, because we can imagine a game being played between the adversary and the model builder.

Information-theoretic lower bounds

General idea: To show a problem is hard, show there are many possible outputs. The algorithm must do a lot of work to distinguish the correct one.

Example: Sorting by comparisons requires $\Omega(n \log n)$ comparisons on average.

Let $X_1, \ldots, X_n$ be keys to be sorted. We assume they are given in random order (each permutation equally likely).

We consider algorithms that use queries of the form "is $X_i < X_j$?"

Such algorithms can be specified by *decision trees*. For example, here is a decision tree for sorting the keys a, b, c.

Note that this has $3! = 6$ leaves, with an average depth of $8/3$.

Since we're interested in lower bounds, let's assume that all redundant comparisons (those for which the answer can already be determined) are eliminated. The resulting tree has the property that each node has 0 or 2 children.

Lemma: Let $x_1, \ldots, x_m$ be numbers with $0 \le x_i \le 1$, and $\sum x_i = 1$. Then

$$g(x_1, \ldots, x_m) := \frac{\sum_{i=1}^{m} (-\log x_i)}{m} \ge \log m.$$

Proof: Use convexity. Note that $f(x) = \log x$ has the property that

$$f(\frac{x+y}{2}) > \frac{f(x) + f(y)}{2}$$

whenever $x < y$. From this, it follows that the minimum value of g, should it exist, occurs when $x_1 = \ldots = x_m$. (Suppose, say, that $x_1 < x_2$. If we replace

both by their average, that reduces the value of g. So no point but this one can yield the minimum.) Why is there a minimum? If we agree that $\log 0 = +\infty$, then g is a continuous function on a closed and bounded set. By a standard result of calculus, such a function is bounded, and must take on its minimum and maximum values.

This says that the average of the values of f is bounded below by f of the average. The generalization of this is a standard, useful result called *Jensen's inequality*. This says that when f is convex and X is a random variable, $E[f(X)] \geq f(EX)$.

An lower bound for sorting

Consider any algorithm that sorts by comparing keys. Its decision tree (pruned) will have $m = n!$ leaves. (Any two leaves must correspond to distinct orderings, since they must differ for the comparison corresponding to their least common ancestor.)

Let $x_i = 2^{-h_i}$, where h_i is the distance of leaf i to the root. We have

$$\sum x_i = 1$$

(start at the root and flip a fair coin to decide which branch to take). SInce $h_i = -\log_2 x_i$, our lemma gives us

$$\text{average leaf depth } = \frac{\sum -\log_2 x_i}{n!} \geq \log_2 n!.$$

Apply Stirling's formula, and conclude that we need

$$\geq n \log_2 n - O(n)$$

key comparisons to sort, on average.

Can this be attained?

It can be shown that Mergesort uses $n \log_2 n + O(n)$ key comparisons.

Why is this called an information theory bound?

Suppose we observe the algorithm, recording the results of the comparisons. This gives us a sequence of bits that encode an ordering (namely, the ordering of the inputs).

Example: for the decision tree we exhibited for $n = 3$ the possible bit sequences (code words) are

$$00,\ 101,\ 011,\ 10,\ 110,\ 111.$$

Our bound for sorting implies that the code words must have average length $\geq \log_2 3! = 2.584963....$ Note that the actual average length is $8/3 = 2.666666...$, so this is respectable.

Lecture 47

Topic du jour: Proof of entropy bound, applications to sorting, merging, etc.

Lower bounds on average code length

Let $\alpha_1, \ldots, \alpha_m$ be words over the binary alphabet $\{0, 1\}$. Assume that they form a *prefix code* in the sense that no word is a prefix of any other one.

If we assign α_i the probability p_i, then

$$\sum_{i=1}^{m} p_i |\alpha_i| \geq H(p_1, \ldots, p_m),$$

where H is the *entropy*

$$H(p_1, \ldots, p_m) := -\sum_{i=1}^{m} p_i \log p_i.$$

We interpret this result as follows. Suppose we have m messages to send, and we encode them by the words $\alpha_1, \ldots, \alpha_m$. Then the average message length is bounded below by the entropy of the distribution on messages.

The term entropy comes from statistical physics. See, e.g. R. P. Feynman, *Statistical Mechanics*, p. 6.

Our interest in this result is that the the paths through a decision tree form a prefix code.

Proof of the lower bound

We will do this in two stages. First, we construct an optimal code. (This is of independent interest.) Then we relax the discrete problem to a continuous one and show that H gives a lower bound on the average code word length.

Constructing optimal codes

Suppose we are allowed to choose the code words. How short can we make the average message length?

Huffman (1952) gave the following construction for an optimal code. Choose the two smallest probabilities, say p and q. Recursively solve the problem for the probabilities $\{p'_1, \ldots, p'_{m-1}\}$ obtained by replacing p, q by $p + q$. Add 0 (resp. 1) to the code word for $p + q$ to get the code word for p (resp. q).

This is easiest to view in terms of a tree. Suppose, for example, the probabilities are 0.1, 0.1, 0.2, 0.3, 0.3. We obtain the tree

Why is this an optimal code?

We give a physical argument. Imagine that the probabilities represent masses. We give each a potential energy equal to its distance from the root in the tree T. (Think of the tree as growing upward, against gravity.) Our job is to put them at leaves of the tree so as to minimize the total potential energy.

The two lightest masses must be at the farthest distance from the root. (If not, we could swap one of them with a mass at this distance and reduce the energy.) We may as well make them the two children of a common parent. Now delete p and q, and give their combined mass $p+q$ to the parent. This produces a new tree T', and the potentials satisfy

$$P(T) = (p+q) + P(T').$$

By induction, T' is an optimal tree, so the same is true of T.

The entropy bound

So far we have assumed unit edge lengths. What if we allowed them to be fractions? Could we reduce the potential even further?

There has to be a constraint (otherwise the whole tree collapses), which we take to be

$$\sum_x 2^{-\text{height}(x)} = 1.$$

(This is supposed to hold for any subtree, and you can see that it suffices to have it hold between parents and children.)

Let's try to mimic Huffman's algorithm. Our first step will be to place the lightest masses p and q at distances x and y from their parent. We are led to the optimization problem

$$\min\{px + qy : 2^{-x} + 2^{-y} = 1,\ x, y > 0\}.$$

This can be solved using Lagrange multipliers. The Jacobian of the equations

$$\begin{aligned} px + qy &= \min \\ 2^{-x} + 2^{-y} &= 1 \end{aligned}$$

vanishes when

$$\begin{vmatrix} p & q \\ 2^{-x} & 2^{-y} \end{vmatrix} = 0,$$

i.e. if

$$x - y = \log_2(q/p).$$

Combining this with the other equation

$$2^{-x} + 2^{-y} = 1$$

and solving, we find

$$x = \log_2(q/p + 1) = -\log_2(p/(q+p)),$$
$$y = \log_2(p/q + 1) = -\log_2(q/(p+q)).$$

Apply the recursive construction as before, but now with these fractional lengths. In replacing the leaves p and q by the new leaf q, the potential drops by

$$\begin{aligned} px + qy = & -p\log_2(p/(q+p)) - q\log_2(q/(p+q)) \\ = & -p\log_2 p - q\log_2 q + (p+q)\log_2(p+q). \end{aligned}$$

By induction, the potential of the resulting tree is

$$-\sum_{i=1}^{m} p_i \log p_i.$$

By considering the entropy of $p/(p+q), q/(p+q)$, you can verify that

$$p + q \geq px + qy$$

at every step. Therefore the tree constructed this way has potential no larger than what we would get by using Huffman's algorithm.

Since the Huffman tree was itself optimal for prefix codes, we get

$$\sum_{i=1}^{m} p_i|\alpha_i| \geq H(p_1, \ldots, p_m)$$

For prefix codes from a q-letter alphabet we have the corresponding bound

$$\sum_{i=1}^{m} p_i|\alpha_i| \geq \frac{H(p_1, \ldots, p_m)}{\log_2 q}.$$

Applications

Recall the main idea: The paths through a decision tree form a prefix code, whose average length must satisfy the entropy bound. This gives a lower bound on the average number of queries asked by *any* algorithm.

Sorting with equality.

We want to sort $X_1, \ldots, X_n$, and we are allowed to ask questions of the form "is X_i greater, equal to, or smaller than X_j"?

The output of such an algorithm will be a ranking with ties, for example

$$X_2 < X_4 < X_1 = X_3.$$

We proved that the number of possible outputs is

$$\sim \frac{n!}{2(\log 2)^{n+1}}$$

(see Lecture 20, 3/8/96). The uniform distribution on these outputs has entropy

$$\log_2 n! + O(n) = n \log_2 n + O(n).$$

Therefore, the average number of queries (assuming the uniform distribution for inputs) used by any algorithm must be at least

$$\frac{n \log_2 n + O(n)}{\log_2 3} = n \log_3 n + O(n).$$

Of course, this gives a bound on the worst case too.

A related question: suppose we draw i.i.d. inputs from the discrete set $\{1, \ldots, N\}$. What happens then? Knuth (v. 3, p. 11) gives asymptotics for this case as an exercise.

Merging

In this problem we are given inputs

$$X_1 < X_2 < \cdots < X_n$$

and

$$Y_1 < Y_2 < \cdots < Y_n$$

to be merged into a sorted list.

There are $\binom{2n}{n}$ possible outputs. (Proof: We can choose places for the X_i's in that many ways, and then the Y_i's are determined.)

By Stirling's formula

$$\binom{2n}{n} = \frac{4^n}{\sqrt{\pi n}},$$

so the uniform distribution on these outputs has entropy

$$\geq 2n + O(\log n).$$

The average number of comparisons must be at least this large (for our input model).

Remark: There is an algorithm that uses $\leq 2n-1$ comparisons. The idea is to repeatedly remove the maximum element, which has to be at the head of one of the lists.

Binary search

The input is a sorted list

$$Y_1 < Y_2 < \cdots < Y_n$$

plus an additional element X. We want to locate X in the list or prove it is not there, using queries of the form "is $X \leq Y_i$"?

Input model: Assume that with probability p, X is in the list (and uniformly distributed), and that with probability q $(= 1-p)$, X is uniformly distributed in one of the $n+1$ "gaps" formed by the Y_i's.

This is an example of a *mixture* distribution. Its entropy is

$$\begin{aligned} H = &\ n\left(-\frac{p}{n}\log_2\frac{p}{n}\right) + (n+1)\left(-\frac{q}{n+1}\log_2\frac{q}{n+1}\right) \\ = &\ p\log_2\frac{n}{p} + q\log_2\frac{n+1}{q} \\ = &\ \log_2 n + H(p,q) + q\log(1+1/n). \end{aligned}$$

There is another way to get this result that may be more instructive. It can be shown that for any random variables U and V we have $H(U) = H(V) + H(U|V)$, where $H(U|V)$ is the average entropy of the conditional distributions for U given $V = v$. If V indicates whether X is in the list or not, then

$$H(V) = H(p,q).$$

In either case we have

$$H(U|V=v) \geq \log_2 n.$$

So the input entropy is $\geq \log_2 n + H(p,q)$.

Therefore (under our model) the average number of comparisons used is at least $\log_2 n + O(1)$.

Lecture 48

Topic du jour: Lower bounds for searching in partial orders

Saks and Linial, J. Algorithms 6, 86-103, 1985. See R. P. Stanley, Enumerative Combinatorics, Chapter 3 for background on partial orders.

Partial orders

These are structures $(P, \leq)$ satisfying the following axioms:

1. For all x, $x \leq x$.
2. For all x, y, z, if $x \leq y \leq z$, then $x \leq z$
3. For all x, y, if $x \leq y$ and $y \leq x$, then $x = y$..

We will write $x < y$ as an abberviation for $x \leq y$ and $x \neq y$.

Examples:

The integers, under the usual $\leq$ relation.

The natural numbers $\{1, 2, \ldots\}$, under divisibility.

Any acyclic directed graph, with $x \leq y$ iff there is a directed path from x to y. (For this reason, it's common to intermix graph-theoretic and partial order terminology.)

Finite partial orders are often presented by means of a *Hasse diagram*. This is a graph with vertex set P, and edges $x \to y$ when $x < y$ and there is no z with $x < z < y$. (In this case we say that x *covers* y.)

It's traditional to write these with larger elements above smaller ones, for example

Search problems

We want to study an abstract version of the search problem. Suppose we have a partial ordering P, and each element p of P is assigned a number. (We'll call this the *value* at p, and write it $v(p)$.) Suppose x is some other number. We are allowed to choose elements of P, and ask

" is $v(p)$, greater than, equal to, or less than x? "

Example: $P = \{1, 2, \ldots, n\}$ with the usual ordering. This corresponds to searching an ordered table.

To get a lower bound, note than any proof that $x \notin P$ has to separate the elements of P into two disjoint sets:

$$L_x = \{p : v(p) < x\}$$

and

$$U_x = \{p : v(p) > x\}$$

(Each element has to be directly compared to x, or ruled out by transitivity.)

The set L_x is an *ideal*. (Defn: I is an ideal if whenever $p \in I$ and $q \leq p$, then $q \in I$.) Its complement U_x is a *filter*. (Defn: F is a filter if $\bar{F}$ is an ideal.)

Let $I(P)$ denote the number of distinct ideals of P. We note that each ideal can occur as the result of a search problem. (Make $v(p) = 0$ for all $p \in I$ and $v(p) = 1$ for $p \notin I$. Then set $x = 1$.) Also, different branches of the decision tree for searching correspond to different ideals (since they must classify at least one element differently). We therefore have the information-theory bound:

$$\#\text{ of queries } \geq \log_2 I(P).$$

If $x \notin P$, then each query has two possible answers. For this reason, we use binary logarithms.

Strictly speaking, this is a bound on the number of queries needed in the worst case to prove that $x \notin P$.

It's also an average-case bound, if we take as our input model that all ideals are equally likely.

Examples of search problems

Example 1: Searching an unordered set of size n.

We take $|P| = n$, with the vacuous $\leq$ relation. (That is, no pair of elements is comparable.)

Then every subset is an ideal, so $I(P) = 2^n$. We have

$$\log_2 I(P) = n,$$

so n queries are required to prove $x \notin P$.

This is intuitively obvious, since we have to examine every element. (Any unexamined element might have the value x.)

Example 2: Searching a linear order.

We let P be the linear order on n elements, for example $\{1, \ldots, n\}$ with the usual ordering.

The ideals are $\emptyset$, together with the sets

$$I_x := \{y : y \leq x\},$$

so

$$I(P) = n + 1.$$

Therefore, we need $\geq \log_2(n+1)$ queries to verify that $x \notin P$. This agrees with our previous analysis of binary search, if we take $p = 0$ and $q = 1$.

Example 3: Searching a 2-dimensional array.

We are given an $m \times n$ array of numbers, arranged so that the values are nondecreasing along rows and columns. For example, if $m = 3$ and $n = 4$, we might have

$$\begin{pmatrix} 3 & 13 & 29 & 30 \\ 2 & 10 & 20 & 21 \\ 0 & 9 & 17 & 18 \end{pmatrix}$$

To fit this into our theory, let

$$P = \{(i, j) : 1 \leq i \leq m, 1 \leq j \leq n\}$$

and put

$$(i, j) \leq (i', j') \iff \begin{cases} i \leq i', & \text{if } j = j'; \\ j \leq j', & \text{if } i = i'. \end{cases}$$

This corresponds to a lattice L with mn vertices:

What are the ideals? Consider the "dual lattice" – I'm not sure if that is the correct term – $\hat{L}$ obtained from L:

Observe that each monotonic path from A to B determines an ideal, and that every ideal arises in this way. (Points of L lying "southwest" of the path belong to the ideal.)

From our previous work, we know there are $\binom{m+n}{n}$ lattice paths, so

$$I(P) = \binom{m+n}{n}.$$

The theory tells us that we need

$$\leq \log_2 I(P) = \log)_2 \binom{m+n}{n}$$

queries in the worst case, to prove that $x \notin L$.

Connections to merging

We note that the number of ideals in an $m \times n$ lattice is the same as the number of ways to merge

$$X_1 < X_2 < \cdots < X_m$$

with

$$Y_1 < Y_2 < \cdots < Y_n.$$

In fact, there are 1-1 correspondences:

$$\text{ideals} \leftrightarrow \text{lattice paths} \leftrightarrow \text{merge results}$$

We have already seen how to identify ideals with lattice paths. We illustrate the other correspondence with an example. Consider a path such as

This corresponds to

$$X_1 < Y_1 < Y_2 < X_2 < Y_3 < Y_4 < X_3 < Y_5$$

Even more is true: There is a perfect correspondence between algorithms that search the lattice and algorithms that perform merging.

If we choose coordinates so that i increases as we go down, and j increases across, then

$$A_{ij} < x \iff \text{Path goes "northwest" of } (i,j) \iff Y_j < X_i.$$

This allows us to derive an optimal searching algorithm. We already know an optimal merging algorithm, which uses $2n-1$ comparisons to merge two lists of size n. (Actually we didn't show it was optimal, but that is true.) The algorithm repeatedly compares the smallest of the remaining X_i's and Y_j's, removing it from the list it is on. The analogous step in searching is to compare x to the upper left corner of an array such as

$$\begin{pmatrix} 3 & 13 & 29 \\ 2 & 10 & 20 \\ 0 & 9 & 17 \end{pmatrix}.$$

With this query, we either find x, or remove one row or one column from contention. The number of such removals that can be done before the array vanishes is $2n-1$.

Notes

Efron dice problem (given two dice, figure out the number of ways they can come up less, equal, more) should also be isomorphic to merging. Is it?

Lecture 49

Topic du jour: Lower bounds for searching in partial orders (continuation)

References: Saks and Linial, J. Algorithms 6, 86-103, 1985. Carlsson, Proc. 2nd SODA, 1993.

Review of last lecture

Let $(P, \leq)$ be a partial order.

An *ideal* in P is a set that is closed under "going down." More precisely, J is an ideal if

$$p \in J, q \leq p \qquad \Longrightarrow q \in J.$$

Saks-Linial bound: The cost to search for an element in P is at least $\log_2 I(P)$, there $I(P)$ denotes the number of ideals in P.

We applied this last time to unordered sets, ordered lists, and 2-dimensional arrays. Today we will consider heaps.

Heaps

Let T be a full binary tree with nodes given real values. We call T a *heap* if for every node x,

$$v(x) \leq v(x\text{'s parent})$$

Here's an example:

For full trees, we let n indicate the number of nodes and d the depth (maximum root-leaf distance). The above tree then has $n = 7$ and $d = 2$. In general,

$$n = 2^{d+1} - 1.$$

A tree is a partial ordering under the ancestor relation. To say that the tree is a heap means that it has been labelled in a way consistent with that relation.

How many ideals in a tree?

A recurrence relation

Let I_d denote the number of ideals in a full tree of depth d. Then $I_0 = 2$ (there are only the trivial ideals $\emptyset$ and P), and for $d \geq 1$,

$$I_d = 1 + I_{d-1}^2.$$

This is because ideal must either be the whole tree, or the union of two ideals taken from the left or right subtrees.

This recurrence relation has been studied in another context. (See Aho and Sloane, Fibonacci Quarterly, 1973). We have

$$I_d = \lfloor k^{2^{d+1}} \rfloor,$$

where

$$k = \exp\left(\sum_{j \geq 0} \frac{1}{2^{j+1}} \frac{I_j}{I_j + 1}\right) = 1.502837...$$

Therefore,

$$\log_2 I_d \sim 2^{d+1} \log_2 k \sim n \log_2 k.$$

We note that $\log_2 k = 0.587688...$, so no heap searching algorithm can examine fewer than 58% of the nodes in a full tree.

What's the best heap searching algorithm?

Examining all elements uses n queries.

We can do better by observing that most of the nodes lie near the bottom of the tree.

To find x, first search the next-to-last row. For each such node q, if $q > x$, eliminate its ancestors, and if $q < x$, eliminate its two children. Search all remaining keys.

Complexity?

Since $n = 2^{d+1}$, the next-to-last row has 2^{d-1} nodes. Label each of these with $+$ if it is $> x$, and $-$ if it is $< x$. Each $+$ key contributes two bottom row keys to the next part of the search, and the number of keys above the next-to-last row that remain cannot exceed the number of $-$ keys. (To see this, note that a key is knocked out unless all of its descendants in the next-to-last row are labelled $-$. Using these descendants, we can partition the remaining upper keys into disjoint full subtrees.) Hence the number of remaining keys cannot exceed

$$2 \times \text{ size of next-to-last row } = 2^d.$$

The total number of keys examined is

$$\leq 2^{d-1} + 2^d \sim \frac{3n}{4}.$$

Is $3n/4$ best possible?

There is a simple argument to show that the algorithm of the last section is best possible.

Let $x = 1/2$, and label the tree with 0 at the bottom row, and 1 everywhere else. If there is an element in the last two rows that is not examined, then it is consistent with the other data to make its value 1/2. So all of these keys must be queried.

This shows that the Saks-Linial bound is not always optimal.

Another priority queue model

Suppose we impose the requirement on a full binary tree that the keys must be ordered by level. That is, any key at distance i from the root must be greater than or equal to keys at distance $i + 1$.

This is equivalent to labelling a partial order made from disjoint sets $S_0, S_1, \ldots, S_{k-1}$, where $|S_i| = 2^i$, and $x \in S_i \geq y \in S_j$ iff $i < j$. An example is

$$\{10\} \geq \{8, 6\} \geq \{1, 4, 2, 3\}.$$

What are the ideals? These are the empty set, plus those determined by nonempty subsets of some S_i. (Note that if an ideal contains an element of S_i, then it contains all of S_j for $j > i$.) So we have

$$I(P) = 1 + \left(2^{2^0} - 1\right) + \left(2^{2^1} - 1\right) + \cdots + \left(2^{2^{k-1}} - 1\right) \sim 2^{2^k}.$$

Therefore,

$$\log_2 I(P) \sim 2^{k-1} \sim n/2,$$

since P has $n = 2^k - 1$ elements.

What's the best algorithm for searching this structure? I don't know the answer, but note that you can always use a heap searching algorithm, since this is just a special case of a heap. Therefore, the best algorithm must search between 50% and 75% of the keys.

Is $\log_2 I(P)$ best possible?

We have seen that the answer is no, as far as exact complexity is concerned. But it is correct up to constant factors.

Linial and Saks [JCTA 1985] proved that there is a $\delta > 0$ such that in every finite partial order, there is an x such that

$$\delta I(P) \leq \#\text{ of ideals with } x \in I \leq (1 - \delta) I(P).$$

(In fact, one can take $\delta = (3 - \log_2 5)/4 = 0.169518....$)

This implies that every partial order can be searched using

$$\leq \frac{\log_2 I(P)}{-\log_2(1-\delta)} < 4 \log_2 I(P)$$

queries.

In some cases this can be improved. For example, Faigle et al. [SIAM J. Comput. 1986] proved that when P is a tree we can take $\delta = 1/3$.

The proofs of these results are not constructive, so they do not easily lead to searching algorithms.

Lecture 50

Topic du jour: Geometric ideas for lower bounds.

References: D. Dobkin and R. J. Lipton, On the Complexity of Computations under Varying Sets of Primitives, J. Comput. Sys. Sci., v. 18, 1979, pp. 86–91.

Overview

Our goal is to generalize the lower bound results for comparisons to algorithms involving *linear queries*:

$$\text{“is } \sum_{i=1}^{n} a_i x_i \le b\text{?”}$$

Here $x_1, \ldots, x_n$ are the inputs to the algorithm, which we think of as a point in n-dimensional space.

In brief, we will show that algorithms that determine if the input belongs to a “complicated” subset of $\mathbf{R}^n$ must use a large number of these queries.

Geometric background

Suppose a set is defined by linear queries. What properties must it have?

Convexity

Defn: Let $S \subset \mathbf{R}^n$. We call S *convex* if for every $x, y \in S$, the line segment connecting x to y lies entirely within S.

An example and non-example of convex sets in $\mathbf{R}^2$:

Analytically, S is convex if

$$\{tx + (1-t)y : 0 \le t \le 1\} \subset S.$$

Any set of the form $\{x : \sum a_i x_i \le b\}$ is convex.

To prove this, suppose that $\sum a_i x_i \le b$ and $\sum a_i y_i \le b$. Then whenever $0 \le t \le 1$, we have

$$\sum a_i(tx_i + (1-t)y_i) = t\sum(a_i x_i) + (1-t)\sum(a_i y_i) \le tb + (1-t)b = b.$$

Such sets are called *closed half-spaces*. The same result holds for the open half-space

$$\{x : \sum a_i x_i < b\},$$

as well as for the complements of these sets.

Any intersection of convex sets is also convex.

This is so because the definition of convexity involves universal quantifiers. If you have never seen this kind of argument before, it is worthwhile to do the requisite symbol-pushing. Assume that S_α is a family of convex sets, with x and y in their intersection. Then

$$\begin{aligned} x, y \in \bigcap_\alpha S_\alpha &\implies \forall \alpha (x, y \in S_\alpha) \\ &\implies \forall \alpha \forall t (xt + (1-t)y \in S_\alpha) \\ &\implies \forall t \forall \alpha (xt + (1-t)y \in S_\alpha) \\ &\implies \forall t (xt + (1-t)y \in \bigcap_\alpha S_\alpha). \end{aligned}$$

so the intersection of the S_α is convex.

Open and closed sets

Defn: A set S is *open* if for each $x \in S$, there is some $r > 0$ for which the ball of radius r around x lies within S.

Complements of open sets are called *closed*.

Warning: A set need not be open or closed. (Example: the interval $(0, 1]$ in $\mathbf{R}^1$.)

Connectedness

Defn: We say that S is *connected* if there do not exist two disjoint open sets whose union contains S.

This definition is awkward to use in practice (you have to look through all pairs of open sets). It can be shown that a weaker property, called *path-connectedness* is sufficient.

Defn: S is path-connected if for every $x, y \in S$, there is a (continuous) path from x to y that lies within S.

For this definition, a path linking x to y is a continuous function $p : [0, 1] : \mathbf{R}^n$ with $p(0) = x$, and $p(1) = y$.

Theorem: Any convex set is connected. (Proof: Take the straight line as the path.)

Theorem: An open set is connected iff it is path-connected.

Connected components

Introduce the following equivalence relation on S: Say that $x \sim y$ whenever there is a connected subset of S that contains both x and y.

The equivalence classes are called *connected components*.

Decision trees for linear query algorithms

Consider any algorithm that decides membership in a set using a finite number of linear queries. We can associate a decision tree with the algorithm, as we usually do with comparisons.

Each leaf of the decision tree has an associated set, namely the set of all inputs that cause the algorithm to reach that leaf.

For linear queries, these are connected convex sets.

Why convex? The set for a leaf is the intersection of half-spaces (which are always convex) defined by the queries.

Why connected? Convex sets are always connected.

This suggests we can prove that a decision problem is hard by showing that its set of "yes" instances has a lot of connected components.

The element distinctness problem

Given $x_1, \ldots, x_n \in \mathbf{R}$, is there a pair $i < j$ with $x_i = x_j$?

Straight enumeration from the definition uses $\binom{n}{2}$ comparisons. We can reduce this to $n \log n + O(n)$ by sorting.

Let's consider the set of "no" inputs, i.e.

$$S := \{(x_1, \ldots, x_n) \in \mathbf{R}^n : \forall i \neq j (x_i \neq x_j)\}$$

What are its connected components?

Fact: The connected components of S are the sets

$$S_\sigma := \{x : x_{\sigma(1)} < x_{\sigma(2)} < \cdots < x_{\sigma(n)}\}$$

where σ ranges over the permutations of n.

Clearly every $x \in S$ belongs to one of these sets. (We can sort!)

Each S_σ is connected, indeed convex. (It is determined by finitely many inequalities.)

On the other hand, $S_\sigma \cap S_\tau$ is empty when $\sigma \neq \tau$. So we have expressed S as the union of pairwise disjoint connected sets, which have to be the components.

The set S has $n!$ connected components, so any decision tree for S (even one with linear queries) must have $\geq n!$ leaves, hence a path of length

$$\geq \log_2 n! = n \log_2 n + O(n).$$

Sorting achieves this bound.

An average-case lower bound for element distinctness

We assume that the x_i are i.i.d. from the uniform distribution on $[0,1]$. Thus, our input space is the unit cube $[0,1]^n$.

Decomposing the cube

Let

$$Y_\sigma := \{x \in [0,1]^n : x_{\sigma(1)} < x_{\sigma(2)} < \cdots < x_{\sigma(n)}\}$$

Each Y_σ has volume $1/n!$. To prove this, observe that an elementary swap $x_i \leftrightarrow x_j$ has Jacobian ± 1, and by using, say, bubblesort, we can transform any Y_σ into Y_1. (This is the first time I have found bubblesort to be useful for anything.) So, all Y_σ's have the same volume. Their union equals the entire cube, except for points lying in a set of zero volume. (Any nontrivial linear subspace of $\mathbf{R}^n$ has volume 0.) So

$$\text{volume}(Y_\sigma) = 1/n!.$$

A lower bound on algorithms that determine element distinctness using pairwise comparisons.

Eliminate redundant queries, so that each node of the tree has two children or none.

Consider leaves associated with "no" outputs.

The set of inputs leading to a given leaf is the union of one or more of the Y_σ's. (Take all the permutations consistent with the results of the queries leading to that leaf.) Hence it has volume $\geq 1/n!$.

But we know there are $\geq n!$ leaves. So the set of inputs leading to a given leaf has volume exactly $1/n!$.

Now we compute

$$\begin{aligned} E[\#\text{ of comparisons}] &= \sum_L E[\#\text{ of comparisons}|L]\Pr[L] \\ &= \frac{1}{n!}\sum_L E[\#\text{ of comparisons}|L] \end{aligned}$$

and observe that the right hand side is just the average depth of a leaf. (Here the sum was taken over leaves L of the decision tree.) From our work on sorting, we know that any decision tree with $n!$ leaves must have average leaf depth

$$\geq \log_2 n! = n\log_2 n - O(n).$$

Therefore

$$E[\#\text{ of comparisons}] \geq n\log_2 n - O(n).$$

In this result, we could replace $U(0,1)$ by any other (absolutely) continuous distribution. This is so because if $F(x) = \Pr[X \leq x]$, then $F(X)$ has the uniform distribution on $[0,1]$. Our results, however, relied only on the relative ordering of the samples.

Related results

We need $\geq \log_2(m+n)!$ linear queries in the worst case to determine if $x_1, \ldots, x_n$, $y_1, \ldots, y_m$ are distinct real numbers with

$$\{x_1, \ldots, x_n\} \cap \{y_1, \ldots, y_m\} = \emptyset.$$

This is proved similarly to the lower bound on element distinctness.

What if ties are allowed? The lower bound fails, because if $m = 1$, we can determine that $y \notin \{x_1, \ldots, x_n\}$ with n equality queries.

Lecture 51

Topic du jour: Geometric ideas for lower bounds (continued).

References: Manber and Tompa, J. ACM (1985?)

Overview

The last lecture provided some lower bounds based on the topological idea of connectedness. In particular, we showed that any algorithm using linear queries to decide if n real numbers are distinct must take time $\Omega(n \log n)$.

In this lecture, we will use the idea that any set defined by linear queries must be convex.

The even distribution problem

Let $x_1, \ldots, x_n \in \mathbf{R}$. Let their order statistics be

$$x_{(1)} \leq x_{(2)} \leq \cdots \leq x_{(n)}.$$

We say the points are *evenly distributed* if for each $i = 2, \ldots, n$, $x_{(i)} - x_{(i-1)} \leq 1$.

This is often called the McDonald's problem. Suppose the n points are locations along a turnpike:

If there is a large gap, you can open up a Burger King!

Why last lecture's techniques fail

Let

$$S = \{x : \text{ for } 2 \leq i \leq n, x_{(i)} - x_{(i-1)} \leq 1\}.$$

Its complement is

$$\bar{S} = \{x : \exists i (x_{(i)} - x_{(i-1)} > 1)\}.$$

Both of these sets are connected, even path-connected, in general.

Why is S connected?

Suppose x and x' are two points in S. We can get a path from x to x' by first shrinking x to its mean μ, moving μ to the mean μ' of x', relabeling coordinates if necessary, and then expanding back out to x'.

Here is a picture:

If $n \geq 3$, then $\bar{S}$ is connected.

The trick is to show that S looks like the Cartesian product of $\mathbf{R}$ and a bounded convex set. For example, in $\mathbf{R}^2$ we have:

We can specify any point in S by its mean value

$$\mu := \frac{\sum_{i=1}^{n} x_i}{n}$$

together with

$$(x_1 - \mu, x_2 - \mu, \ldots, x_{n-1} - \mu).$$

(There is no information in the last coordinate, since $\sum(x_i - \mu) = 0$.) The set of all such $(n-1)$-vectors is convex (it's the projection of a convex set), and all the coordinates are bounded in absolute value, by n, say.

At this point, you should rely on your intuition: If you remove something topologically equivalent to a line from $\mathbf{R}^n$, it is still connected provided $n \geq 3$.

Those interested in rigorous proofs can consider adding technical details to the following argument. Any bounded convex set is contractible to a point. Because the Cartesian product of two connected sets is connected, it is therefore enough to show that $R^n - 0$ is connected when $n \geq 2$. This is a standard result (see, e.g. Dugundgi, Topology, p. 110).

A lower bound based on convexity

Theorem: Let $T \subset \mathbf{R}^n$, and suppose there are points $p_1, \ldots, p_r \in T$ such that for every $i < j$, there is a point $q \notin T$ on the line segment joining p_i to p_j. Then any algorithm determining membership in T with linear queries must use $\geq \log_2 r$ such queries.

Proof: We will show that the decision tree for the algorithm must have at least r leaves. Suppose that p_i and p_j are associated with the same leaf of the decision tree. Then the algorithm will report, incorrectly, that $q \in T$. Therefore, the decision tree for T must have a root-leaf path of length $\leq \log_2 r$, that is, at least this many queries must be used on some input.

What does S look like?

We know that S, the set of "yes" instances for even distribution, is (up to a nonsingular affine transformation) of the form

$$\mathbf{R} \times T.$$

It turns out that T is very far from being convex.

For example, let's take a slice of S through the plane $z = 0$. In the remaining variables x, y, the set looks like a sheared Star of David:

We now consider general n. Let $r = (n-1)!$, and consider the r points

$$p_\sigma = (1, \sigma(2), \ldots, \sigma(n)),$$

indexed by permutations σ of $\{2, \ldots, n\}$. We claim that if $\sigma \neq \tau$, then

$$\frac{p_\sigma + p_\tau}{2} \notin S.$$

To verify this, first choose coordinates so that $\sigma = 1$. For some i with $2 \leq i \leq n$, we must have

$$p_\sigma = (1, 2, \ldots, i-1, i, i+1, \ldots, n)$$
$$p_\tau = (1, 2, \ldots, i-1, > i, \geq i, \ldots, \geq n)$$

Therefore,

$$q = \frac{p_\sigma + p_\tau}{2} = (1, 2, \ldots, i-1, > i, > i, \ldots, > i),$$

so there is a gap greater than 1 in the order statistics, making $q \notin S$.

Applying the lemma, we find that $\geq \log_2(n-1) \sim n \log_2 n$ linear tests are needed to determine membership in S.

Up to constant factors, this is best possible. We can sort the keys with $n \log_2 n + O(n)$ comparisons using, say, merge sort. Then for $i = 2, \ldots, n$ we have to test that $X_{(i)} - X_{(i-1)} \leq 1$.

What about more powerful queries?

Can we get faster algorithms by allowing more powerful queries? In a sense, this is trivially true, since we can test if $x \in S$ by using only one query, namely, "is $x \in S$?"

If you allow queries based on arbitrary function values (even smooth functions), funny things can happen. Here is an example (Reingold, JACM 1972).

Consider the decision problem of determining if $y \geq \max\{x_1, \ldots, x_n\}$. With pairwise comparisons, we need to do n comparisons, since each x_i has to be proved to be $\leq y$, and hence must "lose" one comparison. This number also suffices, since we can compute the maximum (using $n-1$ comparisons) and then compare that with y.

Suppose we allow queries based on evaluating elementary functions (algebraic functions plus logs, exponents and iterates of these). Note that if $x_1, \ldots, x_n$ are distinct non-negative integers and $n = 2^k$, then

$$2^{x_1} + \cdots + 2^{x_{n/2}} > 2^{x_{n/2+1}} + \cdots + 2^{x_n}$$

iff the maximum occurs in $\{x_1, \ldots, x_n\}$. (Consider binary notation.) Using this idea recursively, we can determine the maximum using $\log_2 n$ queries.

Taking a hint from what we know about computers, we might hope that this sort of anomaly goes away if we restrict ourselves to the four basic rational operations (addition, subtraction, multiplication, division). In the next lecture we will discuss work of M. Ben-Or, who showed that this is true.

Lecture 52

Topic du jour: Lower bounds for algebraic decision trees.

References: M. Ben-Or, STOC 1983. For topology of real varieties see J. Milnor, Proc. AMS 1964.

Polynomial queries

So far we have proved lower bounds based on linear queries, including (as a special case) comparison tests.

We want to consider more general algorithms. We note that except for bit-fiddling, real computer programs are restricted to executing simulations of the four basic arithmetic operations. The result of a constant number of these will be a rational function of the inputs. Note that

$$\frac{p(x_1, \ldots, x_n)}{q(x_1, \ldots, x_n)} \geq a$$

iff

$$(p \geq aq \text{ and } q > 0) \text{ or } (p \leq aq \text{ and } q < 0)$$

So we may as well restrict ourselves to polynomial queries, of the form

$$p(x_1, \ldots, x_n) \geq 0.$$

Algebraic computation trees

This is the analog of a decision tree for more general algorithms.

It is a labelled binary tree in which each node has ≤ 2 children. Three kinds of vertices:

Simple vertex (1 child). Computes a sum, difference, product, or quotient. The arguments can be inputs, previously computed values, or real constants.

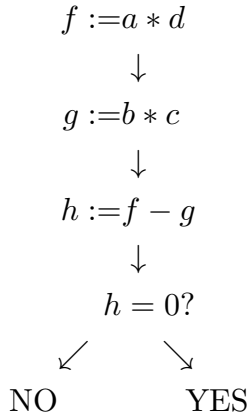
Branching vertex (2 children). Tests if $f > 0$, $f \geq 0$, or $f = 0$, where f is an input or a perviously computed value.

Leaf (0 children). Returns YES or NO, depending on whether x was in the set S.

Here is an algebraic decision tree for determining if the 2×2 system of linear equations

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} u \\ v \end{pmatrix}$$

has a unique solution:



We make the restriction that a division operation is only done when we are certain that the denominator is nonzero.

Bounds for connected components

Defn: An *algebraic set* is a subset of $\mathbf{R}^n$ defined by a finite number of polynomial equations. A *semi-algebraic set* is one defined by a finite number of polynomial equations and inequalities.

Defn: If S is a subset of $\mathbf{R}^n$, we let $\beta_0(S)$ denote the number of connected components of S. (The subscript 0 is traditional, since this is the first of a series of topological invariants called *Betti numbers*.)

We'll use two results of Milnor without proof:

1. If S is semi-algebraic, then $\beta_0(S) < \infty$.
2. If S is an algebraic subset of $\mathbf{R}^n$ defined by polynomial equations of degree $\leq d$, then
$$\beta_0(S) \leq d(2d-1)^{n-1}.$$
We can check this for the Euclidean plane. Bezout's theorem says that two curves of degree d have, in general, $\leq d^2$ intersection points. We note that $d^2 \leq d(2d-1)^{2-1} = 2d^2 - d$.

Our first task is to obtain a bound on the number of connected components of a semi-algebraic set.

Let $S \subset \mathbf{R}^n$ be defined by the following conditions (all polynomials of degree $\leq d$):
$$q_1(x) = \cdots = q_r(x) = 0,$$
$$p_1(x), \ldots, p_s(x) > 0,$$
$$p_{s+1}(x), \ldots, p_h(x) \geq 0.$$
Let $\hat{d} = \max\{d, 2\}$. Then
$$\beta_0(S) \leq \hat{d}(2\hat{d}-1)^{n+h-1}.$$

The proof has three parts.

1. Get rid of strict inequalities. Suppose that S has t connected components. Choose a point in each one, say $v_1, \ldots, v_t$. Let

$$\epsilon = \min_{i \le s, j \le t} \{p_i(v_j)\}.$$

Then the set S_ϵ, defined by

$$q_1(x) = \cdots = q_r(x) = 0,$$

$$p_1(x), \ldots, p_s(x) > \epsilon,$$

$$p_{s+1}(x), \ldots, p_h(x) \ge 0,$$

is a subset of S, with no fewer connected components than S. So we may as well assume that $s = 0$.

2. Introduce slack variables. Note that for real variables,

$$p_j(x) \ge 0 \qquad \Longleftrightarrow \qquad \exists y_j [p_j(x) = y_j^2]$$

Using this idea, we get the new system

$$q_1(x) = \cdots = q_r(x) = 0,$$

$$p_1(x) - y_1^2 = \cdots p_h(x) - y_h^2 = 0,$$

which defines an algebraic set T in R^{n+h}. The degrees of the equations are evidently bounded by $\hat{d}$. By Milnor's theorem

$$\beta_0(T) \le \hat{d}(2\hat{d} - 1)^{n+h-1}.$$

3. Project out slack variables. We observe that the the projection taking R^{n+h} to R^n is continuous. Therefore, since $S_\epsilon = \pi(T)$, we have

$$\beta_0(S_\epsilon) \le \beta_0(T).$$

Why is this? The inverse images of the connected components of S_ϵ must be disjoint relatively open sets (recall that $f^{-1}(\text{open})$ is open, whenever f is continuous), so the number of connected components of T is at least this large.

Computational complexity reflects topological complexity

Theorem: Let S be a subset of $\mathbf{R}^n$, and let T be an algebraic computation tree determining membership in S. Let the tree have height h. Then

$$6^h 3^n \ge \beta_0(S).$$

This can be rewritten as

$$h \geq \log_6 \beta_0(S) - n \log_6 3.$$

To prove this, we define, for each leaf, a semi-algebraic set consisting of the values causing the computation to terminate at that leaf. Each node along the path to the leaf generates one equation, or inequality, possibly with a new variable, as follows:

$f := g \pm h$ generates $y_f = y_g \pm y_h$. (If g and/or h are inputs or constants, we use appropriately labelled x variables, or the constants themselves.)

$f := g * h$ generates $y_f = y_g \pm y_h$.

$f := g \div h$ generates $y_f y_h = y_g$.

$f = 0$? generates $y_f = 0$, similarly for $\geq, >, \leq, <$.

$f \neq 0$? generates $y_f z = 1$, where z is a new variable.

For example, the left branch of our tree for 2×2 linear systems produces the equations

$$y_f = x_a x_b, \qquad y_g = x_b x_c, \qquad y_h = y_f - y_g, \qquad z y_h = 1.$$

You can check that the determinant of $\begin{pmatrix} x_a & x_b \\ x_c & x_d \end{pmatrix}$ is nonzero exactly when there are y_f, y_g, y_h, z satisfying the above equations.

Let V_L be the set of inputs leading to the leaf L. Let U_L be the set of tuples of real numbers satisfying the relations for the path. Evidently V_L is a projection of U_L, so we have

$$\beta_0(V_L) \leq \beta_0(U_L) \leq 2 \cdot 3^{n+h-1} \leq 3^{n+h}.$$

The connected components of V_L lie entirely inside the components of S. (Why? Suppose a component C of V_L had nonempty intersection with two components D, D' of S. Then the subsets $D \cap C$ and $D \cap C'$ are disjoint and relatively open in V_L, so D was not in fact a component.) Summing over all the leaves of the tree, we get

$$\beta_0(S) \leq 2^h \sum_L \beta_0(V_L) \leq 2^h 3^{n+h}.$$

This implies the result.

Applications

Element distinctness: Let

$$S = \{(x_1, \ldots, x_n) : x_i \neq x_j \text{ for } i \neq j\}.$$

We observed earlier that $\beta_0(S) = n!$. Therefore, at least

$$n \log_6 n - O(n)$$

operations are needed to determine membership in S, even in the algebraic tree model.

Union of intervals: This is a 1-dimensional geometric problem. Given $2n$ real numbers $a_1, \ldots, a_n, b_1, \ldots, b_n$, we wish to compute the total length of the set

$$\bigcup_{i=1}^{n} [a_i b_i].$$

If we can do this, we can decide whether

$$\text{length}\left(\bigcup_{i=1}^{n} [x_i, x_i + \epsilon] = n\epsilon.\right)$$

This is true iff the intervals do not overlap except at points, that is, if $|x_i - x_j| \geq \epsilon$ for $i \neq j$. You can verify that this set has $n!$ connected components. Hence, in the algebraic decision tree model, we need

$$\geq n \log_6 n - O(n)$$

operations to compute the length of a union of n intervals.

Notes

For recent work on the Milnor-Thom theorem, see S. Basu, Proc. 1996 STOC. See also R. Fleischer, Proc. 23rd STOC, 1993.

Lecture 53

Topic du jour: Lower bounds via adversary strategies

References: For min-max and second largest, see references in E. Bach, Energy arguments in the theory of algorithms, submitted to AMM. For median, see F. Fich, Lecture Notes, 1986.

Adversary strategies

Most of our lower bounds have been based on showing that a geometric set is complicated in a certain sense. In this lecture, we'll use a different technique entirely.

Suppose an algorithm solves a problem by using a number of queries. Suppose the answers to the queries are controlled by an adversary, whose goal is to delay the algorithm's completion as long as possible. The adversary can give any answer that is consistent with its previous replies.

Often we can prove that the adversary has a strategy that will make the algorithm do a lot of work. This leads to several interesting lower bounds on the complexity of algorithms.

This area is best introduced by means of examples, of which we will do four.

Example: binary search

Suppose we are searching for x in an ordered array

$$A_1 < A_2 < \cdots < A_n.$$

We can ask queries of the form "is $x \leq A_i$?"

At every stage, the set of places where x could be is an interval $[j..k]$. (We'll use Pascal notation here.) Initially the interval is $[1..n]$.

The adversary's rule for answering "is $x \leq A_i$?":

If $i > k$, answer yes.

If $i < j$, answer no.

If $j \leq i \leq k$, choose the answer that makes the remaining interval ($[j..i]$ or $[j+1..k]$) larger.

A lower bound for searching.

After m queries, the interval will contain $\geq n/2^m$ integers. It is impossible for the algorithm to stop unless $n/2^m \leq 1$, that is, unless $m \geq \log_2 n$.

This shows that every searching algorithm must use $\geq \log_2 n$ queries.

Note that this replicates the information theory lower bound. As an exercise, you can show that any algorithm to sort n keys by pairwise comparisons must use $\geq log_2 n!$ comparisons.

Features of an adversary argument:

Constructs *one* bad input for each algorithm.

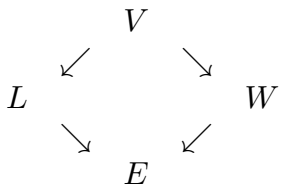
Requires insight to determine the adversary's strategy.

Example: computing max and min

Theorem: Any algorithm determining the minimum and maximum of a set of n keys by comparing them must use at least $3n/2+O(1)$ comparisons in the worst case. This is due to I. Pohl (CACM 1972).

A state diagram

Each key will be in one of the four conditions V (virgin – never compared), W (won but never lost), L (lost but never won), and E (eliminated from contention). Here is a state diagram for these:



Imagine keys (represented by marbles) falling through this diagram. We assign each marble an energy equal to its height, so the marbles in V have energy 2, marbles in W or L have energy 1, and marbles in E have energy 0. The total energy of a configuration is the sum of the energies of all the marbles.

What does the adversary do?

The adversary's rule: Whenever a query is presented, answer it in such a way as to minimize energy loss, but consistent with the answers already given.

Proof of the lower bound

The algorithm must get from the state in which all marbles are in V (energy $2n$) to a state in which L and W have one marble each, with the rest in E (energy 2). Each comparison will drop the energy by an integral amount, equal to 0, 1, or 2. Suppose there are M_i comparisons that drop the energy by i, for $0 \le i \le 2$. Then

$$M_1 + 2M_2 = 2n - 2$$

by energy conservation, whereas

$$n/2 \ge M_2$$

(the adversary can force an energy loss ≤ 1, except for V-V comparisons, of which there can be at most $n/2$). Using these relations, we find that $M_1 + M_2 \ge \frac{3}{2}n - 2$ which gives a lower bound on the number of comparisons.

Can we attain this?

For simplicity we assume that n is even. To reduce the potential energy as quickly as possible, we first do $n/2$ V–V comparisons, then determine the maximum of W and the minimum of L. This uses

$$n/2 + (n/2 - 1) + (n/2 - 1) = 3n/2 - 2$$

key comparisons, which is best possible.

Example: computing median

Let $X_1, \ldots, X_n$ be distinct keys, with the order statistics

$$X_{(1)} < X_{(2)} < \cdots < X_{(n)}.$$

For simplicity let's assume that n is odd. The *median* key is $X_{(\frac{n+1}{2})}$.

It's known that the median can be found with $O(n)$ comparisons [Blum et al., JCSS 1973], whereas clearly $\Omega(n)$ are needed. (We have to look at all the inputs.) What's the optimal constant? The answer is not known.

Using an adversary strategy, we'll show that at most $3n/2$ comparisons have to be made.

We divide the keys into three disjoint sets, called Heaven, Earth, and Hell. Initially all keys start out on Earth. The adversary uses three rules.

1. As you might expect, Heaven > Earth > Hell.

2. Heaven vs. Heaven and Hell vs. Hell queries can be answered in any consistent manner.

3. An Earth vs. Earth query causes these two keys to be "killed," with one sent to Heaven and one to Hell. (We will refrain from any speculations about the theological implications of this.)

These rules are consistent, in the sense that if there is an ordering consistent with the answers before the rule is applied, there is one afterwards.

There is a conservation law:

$$| \text{ Heaven } | \; = \; | \text{ Hell } |.$$

We now observe that if $\hat{x}$ is the median, every other key must be either compared to $\hat{x}$ or ruled out by transitivity. Let us call comparisons of this sort *crucial*.

The adversary forces $\geq (n-1)/2$ non-crucial comparisons to be made. Why is this? Any element on Earth could be the median, since these have never participated in any comparison. Therefore, the algorithm cannot terminate until

$$| \text{ Heaven } | \; = \; | \text{ Hell } | \; = \; \frac{n-1}{2}.$$

Each of these comparisons is non-crucial, so the result follows.

It is not known what the optimal constant is. The best lower bound is due to John W. John [Ph.D., U. Wisconsin, 1986], who showed that at least $2n$ comparisons must be used.

Example: computing the 2nd largest key

Any algorithm that determines the second largest key X_{n-1} must use $\geq n + \log_2 n - 2$ comparisons. This is best possible, as there is a tournament algorithm with this performance.

We note first that any algorithm determining the second largest key must also find the maximum. (To prove it is second largest, it has to be proved not to be the maximum, and the only comparison it can lose to is the maximum.)

We also use a physical model. Think of the n keys as particles, each of which starts off with unit energy. When two keys are compared, the result is a collision that transfers all of the energy to one of the particles, reducing the other's to zero. It is easy to see that a key has positive energy iff it has never lost a comparison.

By conservation of energy, the maximum key ends up with n units of energy. From this, we can see that at least $n - 1$ comparisons must be made to determine the maximum, because every other key must give up its initial energy allotment.

What does the adversary do?

The adversary uses Matthew's rule – "them that gots shall get" – in the following way. When two keys with positive energy are compared, the winner of the comparison is always one with the largest energy. (Ties and queries involving keys of zero energy may be handled in any consistent manner.)

Proof of the lower bound

Suppose the maximum key was compared to k other keys. Then it must have had $\geq n/2$ units of energy before its last comparison, $\geq n/4$ before its next-to-last comparison, and so on. Since its initial energy was 1, we must have $1 \geq n/2^k$, so $k \geq \log_2 n$. Now observe that $k - 1$ of the keys compared to the maximum must lose one additional comparison (since these must also be proved not to be second largest), and the remaining remaining $n - k - 1$ keys must lose at least one comparison. The total number of comparisons is thus bounded below by

$$k + (k - 1) + (n - k - 1) \geq n + \log_2 n - 2.$$

Lecture 54

Topic du jour: Introduction to algebraic complexity theory

References: A. Borodin and I. Munro, The Computational Complexity of Algebraic and Numeric Problems, 1975. S. Winograd, SIAM Book (title?).

What's this about?

The solution of commonplace algebraic problems underlies most of scientific and technical computation. Examples:

Evaluation of polynomials

Multiplication of matrices

Evaluation of linear maps (e.g. Fourier transform).

Our goal is to analyze the work required to solve problems such as these, and prove that all algorithms within a broad class must do at least a certain amount of work.

Some history:

1950's – study of optimal algorithms for polynomial evaluation

1969 – Strassen's recursive algorithm showed that $n \times n$ matrices can be multiplied with $o(n^3)$ operations.

1970's – Investigation of lower bounds based on linear algebra, algebraic geometry, etc.

Straight line programs

We'll consider programs that use the basic arithmetic operations of addition, subtraction, multiplication, and (optionally) division, and do not use branching. Such programs can be written as a sequence of steps, each of which does one operation.

Example: To evaluate $5x^3 + 4x + 1$, we could use the program P below:

$$\begin{aligned} s_1 &:= 3 * x \\ s_2 &:= s_1 - 1 \\ s_3 &:= x * x \\ s_4 &:= x * s_3 \\ s_5 &:= 5 * s_4 \\ s_6 &:= s_5 * s_2 \end{aligned}$$

Note that the i-th statement in such a program is of the form

$$s_i = t \text{ op } u,$$

where t and u are inputs, constants, or references to s_j for $j < i$.

We'll associate a rational function (in the inputs $x_1, \ldots, x_n$) with each assignment in the program. It should be emphasized that these results are purely symbolic.

Example: The program Q, given by

$$\begin{aligned} s_1 &:= x * x \\ s_2 &:= s_1 - 1 \\ s_3 &:= x + 1 \\ s_4 &:= s_3 \div s_2 \end{aligned}$$

computes $x - 1$, even though we could not use the input $x = 1$ in this program.

Why do we do this? One reason is to avoid all the special cases involved in saying precisely how the program behaves. Another is that we are interested in lower bounds, and if we can show that any program that computes a certain function on almost all of its inputs must do a certain amount of work, we have a stronger (or at least no weaker) result.

Evaluation of polynomials

We are given the inputs $a_0, \ldots, a_n$ – think of these as coefficients of a polynomial – and x. We want to evaluate

$$f(x) := \sum_{i=0}^{n} a_i x^i.$$

Straightforward evaluation of this formula uses $2n - 1$ multiplications ($n - 1$ for the powers of x and another n to multiply these by coefficients) and n additions. Can we do better?

Horner's rule

Note that

$$f(x) = \left(\sum_{i=1}^{n} a_i x^{i-1} \right) x + a_0.$$

If we evaluate the expression in parentheses recursively, we use n multiplications and n additions.

Our goal is to show that this is optimal.

Optimality of Horner's rule with respect to $\pm$ operations

Theorem: Any program using $+, -, *$ to evaluate a polynomial of degree n (the inputs are the coefficients and one evaluation point) must use at least n addition/subtraction steps.

We prove this by induction on n. If $n = 0$ the lower bound is trivially true. So, suppose $n > 0$. Our idea is to remove one $\pm$ step, and obtain an algorithm to evaluate

$$g(x) = \sum_{i=0}^{n-1} a_i x^i,$$

which by induction uses at least $n-1$ $\pm$ operations. This will be done by substituting $a_0 = 0$ in the program.

Let z be an input. Let's agree that a step

$$s_i := t \text{ op } u$$

uses z if one of t, u is z, or an s_j with $j < i$ that uses z.

Consider the first $\pm$ step that uses a_n. (There must be one, since the answer depends on a_n.) Suppose it is

$$s_i := q \pm r.$$

Then, either q or r is a nonzero multiple of a_n. (It's either a_n itself, or something that involved a_n only as a multiplicative factor.)

We now modify the given program. First we substitute 0 for a_n, everywhere it appears. The i-th statement now does one of the following three things:

$$\begin{aligned} s_i &:= q \pm 0 \\ s_i &:= 0 + r \\ s_i &:= 0 - r \end{aligned}$$

In the first two cases, we delete this statement and replace s_i by q or r everywhere it appears. In the third case, we replace the statement by

$$s_i := (-1) * r.$$

These modifications yield a new program for evaluating

$$g(x) = \sum_{i=0}^{n-1} a_i x^i,$$

which (by induction) must use at least $n-1$ $\pm$ operations. Since we removed at least one $\pm$ operation to do this, our original program had $\geq n$ $\pm$ operations.

Warning: This says nothing at all about the complexity of evaluating a particular polynomial. For example, the polynomial $x^n - 1$ can be evaluated using only one addition.

Degree bounds

Theorem: If $f(x_1, \ldots, x_m)$ is a polynomial of degree n, any algorithm to evaluate it using $+, -, *$ must use $\geq \log_2 n$ multiplication steps.

The value taken by every s_i is a polynomial, and by induction, $\deg(s_i) \leq 2^i$. (The best we can do at step i is to multiply together two polynomials of degree 2^{i-1}.) If s_k equals f, then

$$n = \deg(s_k) \leq 2^k \Longrightarrow k \geq \log_2 n.$$

This shows that the usual "repeated squaring" algorithm for evaluating powers of x, given recursively by

$$x^n = \begin{cases} x, & \text{if } n = 1; \\ (x^{n/2})^2, & \text{if } n \geq 2 \text{ and } n \text{ is even;} \\ x^{n-1}x, & \text{if } n \geq 3 \text{ and } n \text{ is odd,} \end{cases}$$

is within a constant factor of optimal. As an exercise, you can show that there is an algorithm to evaluate x^n using $(1 + o(1)) \log_2 n$ multiplication steps.

Multiplicative complexity for small n $(= 1, 2)$.

Let's see what the implications of our degree bound are for polynomial evaluation.

As a function of all its variables, the polynomial

$$\sum_{i=0}^{n+1} a_i x^i$$

has degree $n + 1$.

When $n = 1$, $\log_2(n + 1) = 1$, and the usual algorithm to evaluate $ax + b$ uses 1 multiplication.

When $n = 2$, $\log_2(n + 1) = 1.584963...$, and since the number of multiplications must be an integer, we conclude that every algorithm evaluating

$$ax^2 + bx + c$$

uses 2 or more multiplications. For this case, Horner's rule is optimal.

For $n \geq 3$, this method does not show Horner's rule is optimal. For this we need another idea, which (as it turns out) will come from linear algebra.

Lecture 55

Topic du jour: Lower bounds via linear algebra.

References: V. Pan, Russian Math. Surveys 1966; Winograd, Proc. NAS. 1967.

Review of polynomial evaluation

Let $f(x) = \sum_{i=0}^{n} a_i x^i$, considered as a function of its coefficients and evaluation point.

Horner's rule evaluates f using n multiplication operations. For $n = 1, 2$ we know this is best possible.

Our goal is to prove that Horner's rule is optimal for all n.

A lemma from linear algebra

Let F be a field (e.g. $\mathbf{R}$ or $\mathbf{C}$). Let

$$L_1(x_1, \ldots, x_n), \ldots, L_m(x_1, \ldots, x_n)$$

be linearly independent linear forms in the indeterminates $x_1, \ldots, x_n$. (That is, the L_i are linearly independent elements of the dual space $(F^s)^*$.) If we substitute another linear form

$$M(x_2, \ldots, x_n)$$

for x_1 in the L_i's, then at least $m - 1$ of the forms

$$\hat{L}_1 := L_1(M, x_2, \ldots, x_n), \ldots, \hat{L}_m := L_m(M, x_2, \ldots, x_n)$$

are linearly independent.

Proof: The mapping $\psi = (L_1, \ldots, L_m)$ maps F^n onto an m-dimensional space. Also, the mapping

$$\phi(x_2, \ldots, x_n) := (M(x_2, \ldots, x_n), x_2, \ldots, x_n)$$

is a 1-1 mapping from F^{m-1} into F^m. Choose a basis B for the image of ϕ, and add one more vector v to get a basis for F^m. Since

$$\mathrm{Im}(\psi) = \psi(B) + \psi(v)$$

has dimension m by hypothesis, the dimension of $\psi(B)$ must be at least $m - 1$.

Pan's theorem: polynomial evaluation requires n multiplications

Active vs. inactive multiplications

We consider programs to evaluate

$$f(x) = \sum_{i=0}^{n} a_i x^i,$$

given the inputs $a_0, \ldots, a_n, x$.

Defn: Let $s_i := u * v$ be a multiplication step in a program using $+, -, *$. We call this step *inactive* if either u or v is a constant, or if both u and v evaluate to polynomials in x. All other multiplications are called *active*.

A lower bound for active multiplications

In the result below, we write a as shorthand for $a_1, \ldots, a_s$, and $L(a)$ for a linear form in these variables.

Theorem: For every s, and for every polynomial of the form

$$f(a, x) = \sum_{i=1}^{n} L_i(a) x^i + L_0(a) + g(x),$$

at least $r = \operatorname{rank}(L_1, \ldots, L_n)$ active multiplications are required to evaluate f.

Proof: We use induction on s. If $s = 0$, the result is trivial. Suppose $s \geq 1$. Let P be a program computing f. We may as well assume that $r > 0$, because if not, there is nothing to prove. Then there has to be at least one active multiplication, because without them we can only compute functions of the form $L_0(a) + g(x)$. Suppose the first active multiplication computes

$$(L(a) + t(x)) * (M(a) + u(x)).$$

(Here, L and M are linear forms, and t and u are polynomials.) One of the linear forms must be nontrivial, say it is

$$L(a) = \sum_{i=1}^{s} c_i a_i, \qquad c_1 \neq 0.$$

Make the substitution

$$a_1 \leftarrow \frac{-1}{c_1} \left(\sum_{i=2}^{s} c_i a_i + t(x) \right),$$

adding code to evaluate this if necessary. (This can be done with no increase in active multiplications.) Note that after the substitution, the result of the first active multiplication vanishes. Hence we can delete this step, replacing its result by 0 everywhere it is used. This gives us a new program to evaluate

$$\hat{f}(a_2, \ldots, a_s, x) := \sum_{i=1}^{n} \hat{L}_i(a) x^i + \hat{L}_0(a) + \hat{g}(x).$$

By the lemma proved at the beginning of this lecture, $\text{rank}(\hat{L}_1, \ldots, \hat{L}_n) \geq r - 1$. By our induction hypothesis, at least $r - 1$ active multiplications remain, so the original program had $\geq r$ of them.

Corollary (Pan): Every program using $+, -, *$ to evaluate

$$f(x) = \sum_{i=0}^{n} a_i x^i$$

has at least n multiplications.

Proof: Observe that $a_1, \ldots, a_n$ are linearly independent.

This result generalizes to programs using division if the underlying coefficient field is infinite.

Winograd's theorem

Let F be a field of coefficients (e.g. $\mathbf{R}$ or $\mathbf{C}$). Let $x_1, \ldots, x_n$ and $y_1, \ldots, y_m$ be indeterminates. We consider programs to compute

$$(x_1, \ldots, x_n) \begin{pmatrix} \varphi_{11} & \cdots & \varphi_{1m} \\ \varphi_{21} & \cdots & \varphi_{2m} \\ \vdots & & \vdots \\ \varphi_{n1} & \cdots & \varphi_{nm} \end{pmatrix},$$

where the φ_{ij} are polynomials in the y's.

Theorem: If every row of the matrix has a nonconstant function, and the nonconstant functions in each column of the matrix are linearly independent over F, then any program (using $+, -, *$) for this purpose must have $\geq n$ multiplications.

We omit the proof. (It is similar to the proof of Pan's theorem.)

If F is infinite, the result extends to programs using division.

Applications:

1. Dot product. To compute $\sum_{i=1}^{n} x_i y_i$ requires n multiplications.

2. Matrix-vector products. Consider Ax, where $A = (a_{ij})$ is an $m \times n$ matrix. We can write this as

$$(a_{11}, \ldots, a_{1n}, \ldots, a_{m1}, \ldots, a_{mn}) \begin{pmatrix} x_1 & \cdots & 0 \\ \vdots & & \vdots \\ x_n & \cdots & 0 \\ & \ddots & \\ 0 & \cdots & x_1 \\ \vdots & & \vdots \\ 0 & \cdots & x_n \end{pmatrix}.$$

Therefore, evaluation of Ax must use mn multiplication operations.

3. Evaluation of one polynomial. Suppose

$$f(x) = \sum_{i=0}^{n} a_i x^i.$$

If we substitute $a_0 = 0$, we get an algorithm to evaluate

$$\sum_{i=1}^{n} a_i x^i,$$

which equals

$$(a_1, \ldots, a_n) \begin{pmatrix} x \\ x^2 \\ \vdots \\ x^n \end{pmatrix}.$$

This must use at least n multiplications.

4. Complex number multiplication. Here is an example where Winograd's theorem does not give optimal results. Let $x + iy$ and $u + iv$ be complex numbers. Their product can be represented as

$$(x\ y) \begin{pmatrix} u & 0 \\ -v & 0 \\ 0 & v \\ 0 & u \end{pmatrix}.$$

Therefore, at least 2 real multiplications are needed. It can be shown, however, that no algorithm uses less than 3.

Notes

You can find more lower bounds based on linear algebra in chapter 12 of Aho/Hopcroft/Ullman.

Lecture 56

Topic du jour: Evaluating quadratic and bilinear polynomials; tensor rank.

References: Winograd; Borodin-Munro

Quadratic polynomials

Defn: A *quadratic polynomial* is one of the form

$$Q(x) = a + \sum_{1 \le i \le n} b_i x_i \sum_{1 \le i < j \le n} c_{ij} x_i x_j,$$

Here, a, b_i, and c_{ij} are constants belonging to some field F, and $x_1, \ldots, x_n$ are variables.

Defn: A *bilinear form* is one of the form

$$B(x, y) = \sum_{1 \le i,j \le n} a_{ij} x_i y_j.$$

Note that this is a restricted kind of quadratic polynomial.

Scalar vs. non-scalar operations

Let P be a straight line program, composed of statements

$$s_i := s_j \{+, -, *, \div\} s_k.$$

Let f_i be the rational function representing what this statement computes.

We distinguish between *scalar* and *non-scalar* operations.

Scalar operation: any addition, subtraction, or multiplication/division by a constant (element of F).

Non-scalar operation: anything else.

Example.

Let $(x_1, \ldots, x_n)$ and $(y_1, \ldots, y_n)$ be two vectors in $\mathbf{R}^n$. We can evaluate their inner product by means of the *polarization identity*

$$(x, y) = \frac{|x + y|^2}{4} - \frac{|x - y|^2}{4}.$$

The program

for $i := 1$ to n
  $z_i := x_i + y_i$
  $w_i := x_i - y_i$

$s := 0$
$d := 0$
for $i = 1$ to n
    $s := z_i^2 + s$
    $d := w_i^2 + d$
$p := s/4 - t/4$

uses $2n$ non-scalar operations, which are all multiplications.

Why do we make this distinction?

In many algorithms, the non-scalar operations are easier to implement. For example, it is common to multiply or divide by 2, which for floating point numbers is a simple increment or decrement of an exponent.

We're often using these algorithms for complex objects such as matrices. As far as we know, multiplying two matrices requires more work than adding them.

If we are interested in lower bounds, it will not harm us to ignore some of the operations.

Normal forms for evaluating quadratic polynomials

Putting multiplications in standard form

Consider a program P using $\{+, -, *\}$ to evaluate a set of quadratic polynomials $Q_1, \ldots, Q_n$.

Theorem: P can be replaced by an equivalent program P' in which all non-scalar multiplications are of the form

$$\left(\sum a_i x_i\right) * \left(\sum b_i x_i\right).$$

This transformation does not increase the number of non-scalar multiplications.

Proof: Let $R = F[x_1, \ldots, x_n]$ be the ring of multivariate polynomials with coefficients in F. The set

$$I = \{f \in R : f \text{ has no term of degree} < 3\}$$

forms an *ideal* in R. (This is set of polynomials that is closed under addition, subtraction and scalar multiplication, and multiplication by arbitrary other polynomials.)

The idea is to compute modulo I and throw away any terms of degree higher than 3 that are generated. (They will eventually be cancelled, so there is no harm in doing this.) Formally, we replace each program variable s by the triple (s_0, s_1, s_2, s), with the intention of using s_i to contain the degree i part of s. In place of the statement

$$s = t * u$$

we put the sequence

$$\begin{aligned} s_0 &:= t_0 u_0 \\ s_1 &:= t_0 u_1 + t_1 u_0 \\ s_2 &:= t_0 u_2 + t_1 * u_1 + t_0 u_2 \\ s &:= s_0 + s_1 + s_2 \end{aligned}$$

This contains a non-scalar multiplication (indicated by $*$) iff the original statement did.

Note that the constants s_0, t_0, u_0 can in fact be determined at the time this transformation is done. Propogating these constants through the program, we can put it in a kind of standard form.

1. Compute all necessary linear forms $\sum a_i x_i$.
2. Compute all necessary products of these linear forms.
3. Output linear combinations of these products.

Elimination of division

Suppose we enlarge our repertoire to include division.

Consider a program P using $\{+, -, *, \div\}$ to evaluate a set of quadratic polynomials $Q_1, \ldots, Q_n$.

Theorem: If F is infinite, then P can be replaced by an equivalent program P' in which each non-scalar division is replaced by a multiplication, at no increase in non-scalar complexity.

This result is due to Strassen. (Reference?)

The idea of the proof is to compute modulo the ideal I as before, but with two new ingredients.

1. If necessary, change variables by the transformation

$$x_i = x_i' + \lambda_i$$

(here $\lambda_i \in F$), so as to ensure that no denominator vanishes when $x_1 = x_2 = \ldots x_n = 0$.

2. Use the Taylor series

$$\frac{1}{1 + s_1 + s_2} = 1 - s_1 + (s_1^2 - s_2) + \ldots$$

to replace each division by an equivalent multiplication.

Bilinear (noncommutative) algorithms

Let $B_1, \ldots, B_p$ be bilinear forms in the variables $x_1, \ldots, x_m, y_1, \ldots, y_n$, so that

$$B_k(x, y) = \sum_{\substack{1 \le i \le m \\ 1 \le j \le m}} b_{ij}^{(k)} x_i y_j, \qquad k = 1, \ldots, p.$$

Among the algorithms to compute these, we will single out those that do not rely on the commutative law of multiplication. These are called *bilinear* or *noncommutative* algorithms.

Why this restriction?

If we are operating with objects like matrices that do not commute, these are the only algorithms we can use.

Many natural algorithms do not rely on the commutativity of multiplication.

Example: complex number multiplication

Let x, y, u, v be real variables. If $i = \sqrt{-1}$, we have

$$(x + iy)(u + iv) = (xu - yv) + i(xv + yu).$$

Computing the product of two complex numbers (given in rectangular coordinates) is essentially that of evaluating the two bilinear forms

$$f := xu - yv$$

and

$$g := xv + yu$$

Tensor rank

Key idea: for bilinear problems, we have

$$\text{min \# of nonscalar } * \qquad = \qquad \text{rank of coefficient tensor}$$

Suppose

$$B_k(x, y) = \sum_{\substack{1 \le i \le m \\ 1 \le j \le m}} b_{ij}^{(k)} x_i y_j, \qquad k = 1, \ldots, p.$$

The *coefficient tensor* is the $m \times n \times p$ array of coefficients

$$T = (a_{ij}^{(k)}).$$

Example: for complex number multiplication, the tensor is

$$\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \qquad \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

(You should think of these matrices as representing the front and back slices of a $2 \times 2 \times 2$ cube.)

Tensor rank.

We call the tensor *elementary* if there are numbers

$$\alpha_1, \ldots, \alpha_m, \beta_1, \ldots, \beta_n, \gamma_1, \ldots, \gamma_p$$

for which

$$a_{ij}^{(k)} = \alpha_i \beta_j \gamma_k.$$

We'll use the shorthand

$$\left(a_{ij}^{(k)}\right) = (\alpha_1, \ldots, \alpha_m) \otimes (\beta_1, \ldots, \beta_n) \otimes (\gamma_1, \ldots, \gamma_p)$$

Every tensor is the sum of elementary ones. The minimum number is called the *rank*.

Bilinear programs can be put in a similar normal form, in which each multiplication is of the form

$$\left(\sum a_i x_i\right) * \left(\sum b_i y_i\right).$$

From this it is easy to see that the minimum number of non-scalar multiplications needed to evaluate a set of bilinear forms is the rank of the coefficient tensor.

Determination of the rank of a tensor is not an easy problem. As we will see, it depends on the coefficient field. The rank is computable for $\mathbf{R}$ and for $\mathbf{C}$, since there are algorithms for deciding the truth of any first-order existential sentence over these fields.

Example: what is the tensor rank for complex multiplication?

Recall that we want to compute

$$xu - yv \qquad \text{and} \qquad xv + yu.$$

Let's change v to $-v$ to make our tensor look nicer. It becomes

$$T = \begin{matrix} & \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \\ \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} & \end{matrix},$$

which is the sum of three elementary tensors:

$$\begin{matrix} & \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \\ \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} & \end{matrix} + \begin{matrix} & \begin{pmatrix} 0 & -1 \\ 0 & 0 \end{pmatrix} \\ \begin{pmatrix} 0 & -1 \\ 0 & 0 \end{pmatrix} & \end{matrix} + \begin{matrix} & \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} \\ \begin{pmatrix} 0 & 0 \\ -1 & 0 \end{pmatrix} & \end{matrix}$$

Therefore, T has rank ≤ 3, and we can derive an algorithm to compute $(x + iy)(u - iv)$ using three real non-scalar multiplications. We write it in a redundant style so as to make the tensor connection clear:

$$\begin{aligned} A &:= ((x+y)(u+v), 0) \\ B &:= (-xv, -xv) \\ C &:= (-yu, yu) \\ D &:= A + B + C = (xu + yv, -xv + yu) \end{aligned}$$

Computing the rank of T

It's easy to see that T cannot have rank 1. Suppose that

$$T = (\alpha_1, \alpha_2) \otimes (\beta_1, \beta_2) \otimes (\gamma_1, \gamma_2).$$

Without loss of generality we can assume that γ_1, the multiplier for the front slice, equals 1. Then we would have

$$\begin{aligned} \alpha_1\beta_1 &= 1 \\ \alpha_1\beta_2 &= 0 \\ \alpha_2\beta_1 &= 0 \\ \alpha_2\beta_2 &= 1 \end{aligned}$$

and these equations are inconsistent. (The first and last imply that all variables are nonzero, which is incompatible with the middle two.)

Can T have rank 2? Suppose we have

$$T = (\alpha_1, \alpha_2) \otimes (\beta_1, \beta_2) \otimes (\gamma_1, \gamma_2) + (\alpha_1', \alpha_2') \otimes (\beta_1', \beta_2') \otimes (\gamma_1', \gamma_2').$$

The front slice is

$$(\alpha_1, \alpha_2) \otimes (\beta_1, \beta_2) \cdot \gamma_1 + (\alpha_1', \alpha_2') \otimes (\beta_1', \beta_2') \cdot \gamma_1',$$

and we may as well take $\gamma_1 = \gamma_1' = 1$. Writing out the equation for the front slice, we find it is equivalent to

$$\begin{pmatrix} \alpha_1 & \alpha_2 \\ \alpha_1' & \alpha_2' \end{pmatrix} \begin{pmatrix} \beta_1 & \beta_1' \\ \beta_2 & \beta_2' \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix},$$

i.e., $AB = I$. The equation for the back slice can be written

$$\begin{pmatrix} \alpha_1 & \alpha_2 \\ \alpha_1' & \alpha_2' \end{pmatrix} \begin{pmatrix} \gamma_2 & 0 \\ 0 & \gamma_2' \end{pmatrix} \begin{pmatrix} \beta_1 & \beta_1' \\ \beta_2 & \beta_2' \end{pmatrix} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix},$$

so we need to know whether $\begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$ is equivalent to a diagonal matrix.

The characteristic polynomial of this matrix is $X^2 + 1$, so the answer is "yes" over $\mathbf{C}$ but "no" over $\mathbf{R}$. Thus, we have

$$\operatorname{rank}(T) = \begin{cases} 3, & \text{over } \mathbf{R}; \\ 2, & \text{over } \mathbf{C}. \end{cases}$$

We conclude that the minimal non-commutative algorithm for complex number multiplication uses 3 non-scalar multiplications if real coefficients are used, but only 2 if complex coefficients are allowed. It can be shown that the first bound holds for commutative algorithms as well.

Lecture 57

Topic du jour: Nonlinear lower bounds (a la Strassen)

References: V. Strassen, Numer. Math. 1973. See also chapter V of Borodin-Munro. Reference books on algebraic geometry include M. Reid, Undergraduate Algebraic Geometry, and I. Shafarevich, Basic Algebraic Geometry.

Overview

So far, all of our lower bounds for algebraic problems have been linear. Example: To evaluate a polynomial of degree n requires $\geq n$ multiplications.

In 1973, Strassen devised a way to prove nonlinear lower bounds, based on the geometry of the solutions to simultaneous polynomial equations.

Ideas from algebraic geometry

Let $\mathbf{C}$ be the set of complex numbers. Complex n-space is

$$\mathbf{C}^n = \{(x_1, \ldots, x_n) : x_i \in \mathbf{C}\}.$$

A subset of $\mathbf{C}^n$ is called *algebraic* if it is defined by a finite number of polynomial equations

$$p_1(x_1, \ldots, x_r) = p_2(x_1, \ldots, x_r) = \cdots = p_r(x_1, \ldots, x_r) = 0.$$

(Some people say *closed*, or more properly, *Zariski closed*.)

Some examples:

Points (defined by $x_i = a_i$, $i = 1, \ldots, n$.)

Hypersurfaces, defined by one equation $p(x_1, \ldots, x_n) = 0$.

Linear varieties, defined by linear equations $Ax = b$.

A *variety* is an algebraic set that is not the union of two smaller ones.

Example. In $\mathbf{C}^2$, the set defined by $y = x^2$ is a variety. More generally, it can be shown that any set X defined by equations of the form

$$y_1 = p_1(x_1, \ldots, x_n), \ldots, y_r = p_r(x_1, \ldots, x_n),$$

is a variety. Intuitively this is so because X can be smoothly mapped in a 1-1 fashion onto $\mathbf{C}^n$ –just ignore the y's – and there is no way to separate $\mathbf{C}^n$ into two algebraic sets.

Non-example: We have $x^2 - y^2 = (x + y)(x - y)$. Therefore the solution set for $x^2 - y^2 = 0$ is the union of two proper subsets, namely the loci of $x = y$ and $x = -y$. Thus, X is not a variety.

Dimension.

Every variety V has the following property: almost all points (the exceptions lie on a smaller algebraic set) have a neighborhood that is topologically the same as $\mathbf{C}^m$. This number m is called the *dimension* of V.

Examples:

Points have dimension 0.

The set of solutions to a linear equation $Ax = b$ has dimension $n - r$, when A has rank r.

If $p_1, \ldots, p_r$ are polynomials, then the solution set of

$$y_1 = p_1(x_1, \ldots, x_n), \ldots, y_r = p_r(x_1, \ldots, x_n),$$

has dimension n.

Degree.

Suppose X is a variety of dimension r in $\mathbf{C}^n$. It can be shown almost all linear varieties H of dimension $n - r$ intersect X in a finite set. The *degree* of X is

$$\max_{\{H : \#(H \cap X) < \infty\}} \#(H \cap X).$$

Examples:

The degree of any linear variety is 1. (This includes points as a special case.)

A hypersurface defined by $p(x_1, \ldots, x_n)$ has degree equal to $\deg p$.

As it stands, the degree is not evidently computable. We will not worry about this because we only need lower bounds on degree, which can be easily found.

Bounds on degree.

We will use the following intuitively reasonable fact without proof: If V is the intersection of two varieties X and Y, then

$$\deg V \leq (\deg X)(\deg Y).$$

Let's check that this makes sense for $n = 2$. According to Bezout's theorem, if C and D are two plane curves of degrees d and e, then they intersect in at most de points.

Suppose $\pi : \mathbf{C}^n \to \mathbf{C}^m$ is the projection for which

$$\pi(x_1, \ldots, x_m, x_{m+1}, \ldots, x_n) = (x_1, \ldots, x_m).$$

If π maps Y onto X (both are supposed to be varieties), then $\deg X \leq \deg Y$.

Strassen's degree bound.

Let $f_1, \ldots, f_r$ be a set of polynomials evaluated by a program P that uses the operations $\{+, -, *\}$. Consider the variety X defined by

$$y_1 = f_1(x_1, \ldots, x_n), \ldots, y_r = f_r(x_1, \ldots, x_n).$$

Then P uses $\geq \log_2(\deg X)$ non-scalar multiplications.

Proof: Suppose P has t steps. Introduce a new variable y_i, and write down an equation, for each step of the program:

$$s_i := s_j\{+, -, *\}s_k \qquad \Longrightarrow \qquad y_i = y_j\{+, -, \cdot\}y_k.$$

For example, if we evaluated x^8 by repeated squaring, our equations would be

$$y_1 = x^2,\ y_2 = y_1^2,\ y_3 = y_2^2.$$

These t equations define a variety Y in $\mathbf{C}^{n+t}$, which is determined by linear and quadratic equations. We note that the number of quadratic equations equals the number of non-scalar multiplications. Also, X is a projection of Y, gotten by ignoring all variables except the inputs and outputs. So we have

$$\deg(X) \leq \deg(Y) \leq 2^{\#\text{ of quadratic eqns}} = 2^{\#\text{ of nonscalar } *}$$

This implies that we have $\geq \log_2(\deg X)$ nonscalar multiplications.

Lower bounds for algorithms using division can be proved in the same way. For a division step $s_i := s_j/s_k$, we use the equation $y_i y_k = y_j$.

Applications.

Multipoint evaluation.

Let f be a polynomial of degree m. Any algorithm to evaluate $f(x_1), \ldots, f(x_n)$ given the inputs $x_1, \ldots, x_n$ must use $n \log_2 m$ multiplication steps.

Proof: Choose some a for which $f(x) = a$ has m distinct roots. (Using the derivative, we see that at most $m - 1$ values of a have to be avoided.) Define X by

$$y_1 = f(x_1), \ldots, y_n = f(x_n),$$

and intersect this with the linear variety Y determined by

$$y_1 = y_2 = \ldots = y_n = a.$$

Since we can choose the x_i in m different ways, we have

$$\deg(X) \geq \#(X \cap Y) = n^m.$$

Therefore, $\geq \log_2 n^m = m \log_2 n$ multiplications are required.

This generalizes the familiar degree lower bound for evaluating one polynomial.

The theorem tells us that the straightforward method of evaluating each polynomial separately is essentially best possible.

Elementary symmetric functions.

Let $x_1, \ldots, x_n$ be complex numbers. The *elementary symmetric functions* of the x_i are the coefficients of the degree n monic polynomial having these roots.

Theorem: To evaluate the elementary symmetric functions requires $\geq \log_2 n!$ multiplications.

Proof (sketch): Choose numbers $a_0, \ldots, a_{n-1}$ for which

$$f(X) = X^n - a_1 X^{n-1} + a_2 X^{n-2} - \cdots + (-)^n a_n$$

has n distinct zeroes. Then the equations

$$y_1 = \sum_i x_i = a_1,\ y_2 = \sum_{i<j} x_i x_j = a_2,\ \ldots,\ y_n = x_1 \ldots x_n = a_n$$

have $n!$ distinct solutions. This proves that the degree of the variety defined by

$$y_1 = \sum_i x_i,\ y_2 = \sum_{i<j} x_i x_j,\ \ldots,\ y_n = x_1 \ldots x_n$$

is at least $n!$, and the result follows from Strassen's theorem.

There are algorithms to solve this problem using $O(n \log^2 n)$ operations, of which $O(n \log n)$ are nonscalar multiplications. See Borodin and Munro.

Interpolation.

Suppose we are given inputs $x_0, \ldots, x_n$ and outputs $y_0, \ldots, y_n$, and we want the coefficients of the unique degree n polynomial p for which

$$p(x_i) = y_i, \qquad i = 0, \ldots, n.$$

Using Strassen's theorem, we can show that any algorithm solving this problem has to use $\geq (n+1) \log_2 n$ non-scalar multiplication/division operations.

Idea of proof: Fix the coefficients $a_0, \ldots, a_n$ of the polynomial and the outputs $y_1, \ldots, y_n$. This gives a system of polynomial equations with n^{n+1} solutions in general. Thus, the variety associated with this computational problem has degree $\geq n^{n+1}$, from which the result follows.