Lecture 8: recurrence, fast integer and matrix multiplication, Catalan numbers

Main points:

- solving recurrence equation: master theorem, generating function, combinatorial analysis
- Application: divide and conquer algorithms: **fast integer and matrix multiplication (Karatsuba' s algorithm, Strassen's algorithm)**.

  中国理论计算机科学家，在矩阵乘法的快速算法上做出了杰出贡献!
- gray codes
- formula for Fibonacci numbers
- **Catalan numbers （中国数学家：明安图）**

Note: Consult the reading material for detail of this lecture.

1 Master theorem

Divide and conquer method: a simple idea that produces many fundamental algorithms!

1.1 Fast integer multiplication: Karatsuba's algorithm

Based on a simple identity:

$(a+b)(x+y) = ax + ay + bx + by$.

Used in a creative way!

Write $a = a_1 \cdot 2^{n/2} + a_0$ and $b = b_1 \cdot 2^{n/2} + b_0$. The standard way of multiplication would take time $T(n)$ satisfies

$$T(n) = 4T(\frac{n}{2}) + \Theta(n).$$

Q: what is $T(n)$?

Expand, and suitably apply the previous identity, which yields the time it takes is:

$$T(n) = 3T(\frac{n}{2}) + \Theta(n).$$
$$T(1) = \Theta(1).$$

Q: What is $T(n) =$?

LZ: "This method is known as Karatsuba's algorithm, after its discoverer Anatoly Karatsuba, then a 23-year-old student at Moscow State University. Karatsuba was challenged to find a better algorithm when his professor, the distinguished mathematician Andrey Kolmogorov, announced his hypothesis that the grade-school algorithm was asymptotically optimal. It took Karatsuba only a few days to discover his improvement. Further research has established that there is an algorithm formultiplying n-bit numbers that runs in time $\Theta(n \cdot \log n \dot{\log} \log n)$—that is, not quite linear, but better than $n^{1+\epsilon}$ for any $\epsilon > 0$."

1.2 Fast matrix multiplication: Strassen's algorithm

Similar idea, but much more nontrivial!!

$$\begin{pmatrix} A & B \\ C & D \end{pmatrix} \times \begin{pmatrix} E & F \\ G & H \end{pmatrix} = \begin{pmatrix} AE + BG & AF + BH \\ CE + DG & CF + DH \end{pmatrix}$$

In this multiplication, the time $T(n)$ satisfies

$$T(n) = 8T(\frac{n}{2}) + \Theta(n^2).$$

Q: what is $T(n)$?

What German mathematician Volker Strassen discovered is that you can do the following only 7 multiplication to multiply two matrices, saving 1 multiplication:

$$\begin{aligned} P_1 &= A(F - H) \\ P_2 &= (A + B)H \\ P_3 &= (C + D)E \\ P_4 &= D(G - E) \\ P_5 &= (A + D)(E + H) \\ P_6 &= (B - D)(G + H) \\ P_7 &= (A - C)(E + F) \\ AE + BG &= P_5 + P_4 - P_2 + P_6 \\ AF + BH &= P_1 + P_2 \\ CE + DG &= P_3 + P_4 \\ CF + DH &= P_5 + P_1 - P_3 - P_7 \end{aligned}$$

Hence, Strassen obtains,

$$T(n) = 7T(\frac{n}{2}) + \Theta(n^2).$$

Q: what is $T(n)$?

LZ: "It is hard to overstate how amazing Strassen's result was and still seems. Matrix multiplication, and equivalent operations such as Gaussian elimination, had been studied for centuries, perhaps millennia, by the world's greatest mathematical minds. But it was not until 1970 that anyone noticed a way to perform this basic operation in less than cubic time. And any high school student motivated to push sums and products around on scratch paper might have noticed exactly the same thing!

Strassen's discovery set off a furious competition to find similar algorithms with exponents smaller than $\log_2 7$. As of this writing, the asymptotically fastest known algorithm has time complexity $n^{2.37286\ldots}$ [4], though the constant factor implicit in the $\Theta(\cdot)$-notation is so huge as to render that algorithm useless in practice. Nobody thinks that strange exponent is the last word on this problem!"

As of 2025, the best peer-reviewed matrix multiplication algorithm is by Josh Alman, Ran Duan, Virginia Vassilevska Williams, Yinzhan Xu, Zixuan Xu, and Renfei Zhou, and has complexity $O(n^{2.371339})$ [2]. This work is based on several previous work, including Duan et. al.[3], Williams et. al. [5]. **近年来，矩阵乘法的许多新算法有中国理论计算机科学家的杰出贡献!** It is not known whether matrix multiplication can be performed in $n^{2+o(1)}$ time. This would be optimal, since one must read the n^2 elements of a matrix in order to multiply it with another matrix. "

Check this Quanta Magazine [1] article to learn the fascinating story of matrix multiplication! Fast matrix multiplication is still an active ongoing research.

1.3 Master theorem

statement and proof (if time allows).

Theorem 1. *(The Master Theorem) If $b > 1$, and*

$$T(n) = aT(\frac{n}{b}) + \Theta(n^d).$$
$$T(1) = \Theta(1).$$

Then,

- *if $\log_b a < d$, then $T(n) = \Theta(n^d)$;*
- *if $\log_b a = d$, then $T(n) = \Theta(n^d \log n)$;*
- *if $\log_b a > d$, then $T(n) = \Theta(n^{\log_b a})$.*

This is very useful to derive performance of algorithms that use recurrence in its computation.

1.4 Further applicatino: Gray codes

Gray (after Frank Gray) codes: adjacent (binomial code differ by exactly one position) for adjacent numbers.

Application(wikipedia): "Gray codes are widely used to prevent spurious output from electromechanical switches and to facilitate error correction in digital communications such as digital terrestrial television and some cable TV systems. The use of Gray code in these devices helps simplify logic operations and reduce errors in practice."

Q: How much time does it take to generate the n-bit Gray code ?

A: Let $m = 2^n$, then

$$T(m) = T(\frac{m}{2}) + \Theta(m)$$
$$T(1) = \Theta(1).$$

By Master theorem: $T(m) = \Theta(m) = \Theta(2^n)$. So, linear time.

2 Besides the master theorem: the generating function

The generating functionmethod is often useful when the recurrence is an additive relation.

Example: Fibonacci sequence satisfies the recurrence $F_{n+2} = F_{n+1} + F_n$ with starting sequence $1, 1, 2, 3, 5, 8, 13, \ldots$ How to find a formula for F_n?

Suppose $f(n)$ denotes the formula for F_n.

the generating function consists of the following steps:

- Consider a sum
$$S(x) = \sum_{n=1}^{\infty} f(n)x^n.$$
- Multiplying it by x and x^2, respectively, and apply the recurrence relation to derive a function equation for $S(x)$.
- Solve $S(x)$
- Expand $S(x)$ into a sum of infinite series.
- Extract a formula for $f(n)$ as the corresponding coefficient of x^n in the expansion of $S(x)$

For example, for Fibonacci sequence, one would get

$$S(x) = f(1)x + f(x)x^2 + xS(x) + x^2S(x) - f(1)x^2 = x + xS(x) + x^2S(x) \Longrightarrow S(x) = \frac{x}{1 - x - x^2}.$$

One can then expand this into a sum of infinite series, and then extract $f(n)$ from the coefficients.

2.1 Another method

By obvervation one finds that the sequence F_n behaves like a geometric sequence. So, suppose $F_n = cq^n$ for some unknown number c and q. By the recurrence relation we get an equation

$$cq^{n+2} = cq^{n+1} + cq^n$$

Solving $q^{n+2} = q^{n+1} + q^n$ for q. We will get two q's, $q_1 = \frac{1+\sqrt{5}}{2}$ and $q_1 = \frac{1-\sqrt{5}}{2}$. So, we will get two formulas G_n and G'_n. We can then calculate c from the initial terms of F_n. If we find c that works, great, we are done. Otherwise, note that if two functions satisfy the recurrence relation of Fibonacci, then their sum and difference also satisfy the recurrence. So, one could try $G_n - G'_n$ and $G_n + G'_n$. In our case for Fibonaaci sequence, none of G or G' fit the sequence. Fortunately, $G_n - G'_n$ fits Fibonaaci sequence, and we are done. Indeed, consider

$$G_n = cq_1^n, \quad G'_n = cq_2^n.$$

Note that G_n and G'_n both satisfy the recurrence relation of Fibonacci sequence, that is, $G_{n+2} = G_{n+1} + G_n$ and $G'_{n+2} = G'_{n+1} + G'_n$. Hence, if we let $H_n = G_n - G'_n$, then H_n also satisfies the recurrence relation of Fibonacci sequence, that is, $H_{n+2} = H_{n+1} + H_n$. Now, if we let $H_0 = 0$ and $H_1 = 1$, we could solve $c = 1/\sqrt{5}$, and we can check that $H_2 = 1$, $H_3 = 2$, etc. In fact, since H_n has the same initial terms with Fibonacci sequence, and satisfies the same recurrence relation, we must have H_n is the Fibonacci sequence. In other words, we have found a formula for the n-th term in Fibonacci sequence, which is

$$F_n = H_n = \frac{1}{\sqrt{5}} \left(\left(\frac{1+\sqrt{5}}{2} \right)^n - \left(\frac{1+\sqrt{5}}{2} \right)^n \right), \quad n = 1, 2, 3, \ldots$$

3 Combinatorial method

Some recurrence equation does not fit either the master theorem, or the generating function (or solving the equations are more complicated), for such equations, sometimes combinatorial analysis can help.

Example: Catalan number is defined to be the number of different balanced strings of parentheses, with n left parentheses and n matching right parentheses. Surprisingly, the Catalan numbers turn out to count a great many things besides balanced strings of parentheses, check **reading material** to learn other counting problems where Catalan numbers appear.

The first Catalan numbers for $n = 0, 1, 2, 3, ...$ are

$1, 1, 2, 5, 14, 42, 132, 429, 1430, 4862, 16796, 58786, ...$

For example, when $n = 3$, we have the following list of balanced strings of parentheses:

$$((())), \quad (())(), \quad ()(()), \quad (()()), \quad ()()().$$

wikipedia: "The Catalan numbers are a sequence of natural numbers that occur in various counting problems, often involving recursively defined objects. They are named after Eugène Catalan, though they were previously discovered in the 1730s by Minggatu (明安图)."

百度: "明安图（1692—1765年），字静庵，蒙古族正白旗人，清朝数学家和天文学家。"

Proposition 2. *(1) Catalan number satisfies recurrence equation*

$$C_n = \sum_{i=0}^{n-1} C_i C_{n-i-1}$$
$$C_0 = 1.$$

(2) $C_n = \frac{1}{2n+1}\binom{2n+1}{n+1} = \frac{1}{n+1}\binom{2n}{n}$.

证明. (1) sketch of the idea: write a balanced string as $w = (u)v$, and induction on size of u.

(2) sketch of the idea: By the circular arrangement idea. Consider $n+1$ left paranthesis and n right ones, arrange them arbitrarily in a circle. Observe that there must exists a unique left parenthesis that is unmatched. For example, consider $n = 3$, if we arrange as follows: ()())((on a circule, starting with any left parenthesis, we can decide that the penultimate symbol '(' is unmatched, and therefore this arrangment corresponds to the balanced strings: (()()). In sum, there is a one-to-one map between such a circular arrangement and balanced strings of parenthesis. This proves the desired formula. □

Obviously, the second formula gives a easier-to-calculate formula for Catalan numbers, which is roughly (as we discussed earlier)

$$C_n = \frac{1}{n+1}\binom{2n}{n} \approx \frac{2^{2n}}{(n+1)\sqrt{n}} \approx \frac{4^n}{n^{3/2}}.$$

中国数学家的贡献

1、矩阵快速乘法：近年来，矩阵乘法的许多新算法有中国理论计算机科学家的杰出贡献，如，段然，等等……

2、卡特兰数

wikipedia: "The Catalan numbers are a sequence of natural numbers that occur in various counting problems, often involving recursively defined objects. They are named after Eugène Catalan(1814-1894), though they were previously discovered in the 1730s by Minggatu (明安图)."

百度: "明安图（1692—1765年），字静庵，蒙古族正白旗人，清朝数学家和天文学家。"

参考文献

[1] New breakthrough brings matrix multiplication closer to ideal. `https://www.quantamagazine.org/new-breakthrough-brings-matrix-multiplication-closer-to-ideal-20240307/`.

[2] Josh Alman, Ran Duan, Virginia Vassilevska Williams, Yinzhan Xu, Zixuan Xu, and Renfei Zhou. More asymmetry yields faster matrix multiplication. In *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2005–2039. SIAM, 2025.

[3] Ran Duan, Hongxun Wu, and Renfei Zhou. Faster matrix multiplication via asymmetric hashing. In *2023 IEEE 64th annual symposium on Foundations of Computer Science (FOCS)*, pages 2129–2138. IEEE, 2023.

[4] François Le Gall. Powers of tensors and fast matrix multiplication. In *Proceedings of the 39th international symposium on symbolic and algebraic computation*, pages 296–303, 2014.

[5] Virginia Vassilevska Williams, Yinzhan Xu, Zixuan Xu, and Renfei Zhou. New bounds for matrix multiplication: from alpha to omega. In *Proceedings of the 2024 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 3792–3835. SIAM, 2024.