Mathematical formulas are designed to be pretty, not to be suitable for computation

devblogs.microsoft.com/oldnewthing/20130508-00

May 8, 2013

Raymond Chen

When you ask a mathematician to come up with a formula to solve a problem, you will get something that looks pretty, but that doesn't mean that it lends itself well to computation.

For example, consider the binomial coefficient, traditionally written ${}_nC_k$ or $C(n, k)$, and in more modern notation as $\binom{n}{k}$. If you ask a mathematician for a formula for the binomial coefficient, you will get the elegant reply

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

(That took forever to format. I will use the traditional notation from now on purely for typographical expediency.)

This is a very beautiful formula, but it's horrible for actual computation because the factorials will be very expensive and are likely to overflow your integer data type even at low values of n. (So you may as well just use a lookup table.) And the $k!$ in the denominator exactly cancels the first k factors in $n!$, so most of your work in multiplying the numbers together is just going to be undone by the division.

For computation, you're much better off using the recurrence $C(n, k) = C(n - 1, k - 1) \times n/k$. This is the recurrence you learned in high school when you had to calculate binomial coefficients by hand: You start with $1 \cdot x^n$ and then to get the next coefficient, you multiply by the exponent on the x and divide by the current position (starting at 1), then decrement the exponent. For example, let's calculate the binomial coefficients $C(8, k)$.

1	$\cdot x^8$	bring down the 8 and divide by 1 (resulting in 1 × 8 ÷ 1 = 8), then decrement the exponent

8	$\cdot x^7$	bring down the 7 and divide by 2 (resulting in 8 × 7 ÷ 2 = 28), then decrement the exponent
28	$\cdot x^6$	bring down the 6 and divide by 3 (resulting in 28 × 6 ÷ 3 = 56), then decrement the exponent
56	$\cdot x^5$	bring down the 5 and divide by 4 (resulting in 56 × 5 ÷ 4 = 70), then decrement the exponent
70	$\cdot x^4$	bring down the 4 and divide by 5 (resulting in 70 × 4 ÷ 5 = 56), then decrement the exponent
56	$\cdot x^3$	bring down the 3 and divide by 6 (resulting in 56 × 3 ÷ 6 = 28), then decrement the exponent
28	$\cdot x^2$	bring down the 2 and divide by 7 (resulting in 28 × 2 ÷ 7 = 8), then decrement the exponent
8	$\cdot x^1$	bring down the 1 and divide by 8 (resulting in 8 × 1 ÷ 8 = 1), then decrement the exponent
1	$\cdot x^0$	bring down the 0, which makes everything zero

(Am I the only person who calculated binomial coefficients by hand?) Notice that the calculations in the second half are the exact inverse of the calculations of the first half, so you only have to do the computations halfway, and then you can just mirror the rest. This is just another way of seeing that $C(n, k) = C(n, n - k)$.

This technique lets you evaluate $C(50, 7) = 99884400$ without overflowing a 32-bit integer.

Often people will ask for an efficient way of calculating factorials, when in fact they don't really need factorials (which is a good thing, because that would require a bignum package); they are really just trying to evaluate a formula that happens to be expressed mathematically with factorials (because factorials are pretty).

Another place pretty formulas prove unsuitable for computation is in Taylor series. The denominator of a Taylor series is typically a factorial, and the numerator can get quite large, too. For example, $\exp(x) = \Sigma\, x^n/n!$. Instead of calculating the power and factorial at each term, use the recurrence

$$\frac{x^n}{n!} = \frac{x}{n} \cdot \frac{x^{n-1}}{(n-1)!}$$

In compiler-terms, you're strength-reducing the loop.

Of course, another problem is that you are adding large numbers first, and then adding smaller numbers later. From a numerical analysis point of view, you should add the smaller numbers first so that they can retain significance longer.

As an example, consider that you have to add the following numbers: 999, and ten 0.1's, and suppose your floating point format is good to only three significant digits. If you added them largest to smallest, you would get this:

999
999 + 0.1 = 999 (three sigificant digits)
999 + 0.1 = 999 (three sigificant digits)
… and so on …

Your final total will be 999. But if you added the smaller numbers first, then you would get

0.1
0.1 + 0.1 = 0.2
0.2 + 0.1 = 0.3
… and so on …
0.9 + 0.1 = 1
1 + 999 = 1000

By adding the small numbers first, you gave them a chance to accumulate to something meaningful before the big number came along and swamped them.

Remember, the way a formula is written on paper is not necessarily the best way of computing it. (And if the formula was written by a mathematician, it is almost certainly not!)

Raymond Chen

Follow