MAXimal

Added: 9 Sep 2008 19:28
Edited: 20 Sep 2010 18:45

Factorial calculation modulo

Contents [hide]

In some cases it is necessary to consider on some prime modulus $p$complex formulas, which may contain, including factorials. Here we consider the case when the module $p$is relatively small. It is clear that this problem is meaningful only when the factorials included in the numerator and denominator of the fractions. Indeed, factorial p!and all subsequent vanish modulo p, but fractions of all the factors containing p, may be reduced, and the resulting expression has to be different from zero modulo p.

Thus, formally **problem** such. Required to compute n!modulo a prime p, thus not taking into account all the multiple $p$factors included in the factorial. By learning to effectively compute a factorial, we can quickly calculate the value of a variety of combinatorial formulas (eg, Binomial coefficients).

Algorithm

Let us write down this "modified" factorial explicitly:

$$n!_{\%p} =$$
$$= 1\cdot 2\cdot 3\cdot \ldots \cdot (p-2)\cdot (p-1)\cdot \underbrace{1}_{p} \cdot (p+1)\cdot (p+2)\cdot \ldots \cdot (2p-1)\cdot \underbrace{2}_{2p} \cdot (2p+1)\cdot \ldots \cdot$$
$$\cdot (p^2-1)\cdot \underbrace{1}_{p^2} \cdot (p^2+1)\cdot \ldots \cdot n =$$
$$= 1\cdot 2\cdot 3\cdot \ldots \cdot (p-2)\cdot (p-1)\cdot \cdot \underbrace{1}_{p} \cdot 1\cdot 2\cdot \ldots \cdot (p-1)\cdot \underbrace{2}_{2p} \cdot 1\cdot 2\cdot \ldots \cdot (p-1)\cdot \underbrace{1}_{p^2} \cdot$$
$$\cdot 1\cdot 2\cdot \ldots \cdot (n\%p) \pmod p.$$

When such a record shows that the "modified" factorial divided into several blocks of length p(the last block may have shorter), which are all identical, except for the last element:

$$n!_{\%p} = \underbrace{1\cdot 2\cdot \ldots \cdot (p-2)\cdot (p-1)\cdot 1}_{1st} \cdot \underbrace{1\cdot 2\cdot \ldots \cdot (p-1)\cdot 2}_{2nd} \cdot \ldots \cdot \underbrace{1\cdot 2\cdot \ldots \cdot (p-1)\cdot 1}_{p-th} \cdot \ldots \cdot$$
$$\cdot \underbrace{1\cdot 2\cdot \ldots \cdot (n\%p)}_{tail} \pmod p.$$

Total count of blocks is easy - it's just $(p-1)! \bmod p$that you can find software or Theorem Wilson (Wilson) immediately find $(p-1)! \bmod p = p-1$. To multiply the common parts of blocks, the obtained value must be raised by the power mod $p$that can be done for $O(\log n)$operations (see. binary exponentiation , however, you can see that we actually erect minus one to some degree, and therefore the result of will always be either 1or $p-1$, depending on the parity index. Meaning in the last, incomplete block, too, can be calculated separately for $O(p)$. Only the last elements of the blocks, we consider them carefully:

$$n!_{\%p} = \underbrace{\ldots \cdot 1} \cdot \underbrace{\ldots \cdot 2} \cdot \underbrace{\ldots \cdot 3} \cdot \ldots \cdot \underbrace{\ldots \cdot (p-1)} \cdot \underbrace{\ldots \cdot 1} \cdot \underbrace{\ldots \cdot 1} \cdot \underbrace{\ldots \cdot 2} \ldots$$

And again we come to the "modified" factorial, but has a smaller dimension (as much as it was full of blocks, and they were $\lfloor n/p \rfloor$). Thus, the calculation of "modified" factorial $n!_{\%p}$we have reduced due $O(p)$to the computation operations already $(n/p)!_{\%p}$. Expanding this recurrence relation, we find that the depth of recursion is $O(\log_p n)$, total **asymptotic behavior of** the algorithm is obtained $O(p \log_p n)$.

Implementation

It is clear that the implementation is not necessary to use recursion explicitly: as tail recursion, it is easy to deploy in the cycle.

```
int factmod (int n, int p) {
        int res = 1;
        while (n > 1) {
                res = (res * ((n/p) % 2 ? p-1 : 1)) % p;
                for (int i=2; i<=n%p; ++i)
                        res = (res * i) % p;
                n /= p;
        }
        return res % p;
}
```

This implementation works for $O(p \log_p n)$.

5 Комментариев e-maxx Войти

Лучшее вначале Поделиться Избранный

Присоединиться к обсуждению...

nikr0s • 11 месяцев назад

А почему нельзя заранее посчитать все остатки от 1! до (p-1)! mod p и записать их в массив, тогда каждый шаг рекурсии будет выполняться за O(1) - найти число полных блоков по модулю p и домножить на хвостовой кусок, взяв его остаток из массива. В итоге ассимптотика будет O(p + logn)

1 • Ответить • Поделиться ›

KappaZ • год назад

А если надо посчитать по модулю p^k?

1 • Ответить • Поделиться ›

Kartikeya Bhardwaj • год назад

I hav the same doubt as angus young

1 • Ответить • Поделиться ›

Angus Young • 2 года назад

This isn't true. If p is smaller than n (or equal):
n! mod p=(1*2*3*...*p*...*n) mod p=((1 mod p)*(2 mod p)*(3 mod p)*...*(P MOD P)*...*(n mod p)) mod p=0.
If p is greater than n:
n! mod p=((1 mod p)*(2 mod p)*(3 mod p)*...*(n mod p)) mod p
And that is O(n).

1 • Ответить • Поделиться ›

LoveYouGuy's → Angus Young • год назад

Here n!_{%p} means n!/(p^[n/p]) mod p, i.e. without p-factors.

1 • Ответить • Поделиться ›