Chinese Remainder Theorem

Theorem

Let $a, b > 1$ be two positive integers.

1. If a is divisible by b, then $m \bmod a$ determines $m \bmod b$: $m \bmod b = (m \bmod a) \bmod b$.
2. (*Chinese remainder theorem*) If $\gcd(a, b) = 1$ (a and b are relatively prime), then every pair of remainders is possible: for every integers u, v with $0 \le u < a$ and $0 \le v < b$ there exists some m such that $m \bmod a = u$ and $m \bmod b = v$.
3. In general, if $d = \gcd(a, b)$, then a pair u, v (where $0 \le u < a$ and $0 \le v < b$) may appear as $(m \bmod a, m \bmod b)$ if and only if $u \equiv v \pmod d$ (i.e., if $u \bmod d = v \bmod d$).

Proof

The first part we have already discussed. Denote $m \bmod a$ by r. Then $m \equiv r \pmod a$, i.e., the difference $m - r$ is a multiple of a. According to our assumption, a is divisible by b, so $m - r$ is a multiple of b, so m and r have the same remainders modulo b, and this is exactly what we have to prove.

The second part. There are a possible remainders modulo a (from 0 to $a - 1$) and b possible remainders modulo b. Therefore, there are ab possible combinations of remainders. We will show that all these combinations appear as $(m \bmod a, m \bmod b)$ as m ranges over $0, 1, 2, \ldots, ab - 1$, once each. (Note that there are ab values of m in the range $0, 1, 2, \ldots, ab - 1$, so this one-to-one correspondence could happen.)

First let us note that *no pair of remainders can appear twice* for different m_1 and m_2 in the range $0, 1, 2, \ldots, ab - 1$. Indeed, this means that

$$m_1 \bmod a = m_2 \bmod a \quad \text{and} \quad m_1 \bmod b = m_2 \bmod b$$

Then $m_1 - m_2$ is divisible both by a and b. Since a, b are relatively prime (our assumption), then $m_1 - m_2$ is divisible by ab, and m_1 and m_2 have the same remainder modulo ab. But all ab numbers in the range $0 \ldots ab - 1$ have different remainders (equal to the numbers themselves), so it is not possible.

Now the last remark: if we have ab pigeons and ab holes, and no two pigeons live in the same hole, then all holes (obviously) will be occupied. Here pigeons are integers in $0 \ldots ab - 1$, and holes are pairs of remainders u, v with $0 \le u < a$ and $0 \le v < b$. Pigeon m lives in the hole $(m \bmod a, m \bmod b)$. We have explained why no hole can be occupied by two pigeons m_1 and m_2, so we know also that every hole (u, v) is occupied by some pigeon m, i.e., $m \bmod a = u$ and $m \bmod b = v$.

Instead of pigeons and holes we may use our knowledge about Diophantine equations. What is the connection? We have to find m that has remainder u modulo a, i.e., has the form $u + xa$ for some x. At the same time m should be equal to $v + yb$ for some y. We see that we need to find x and y such that $u + xa = v + yb$ (and then denote this common value by m).

This is a~Diophantine equation

$$xa - yb = v - u$$

that has solutions since $\gcd(a, b) = 1$ (the minus sign before yv does not matter since we may replace y by $-y$).

Third part is somehow a combination of the first two. If $u = m \bmod a$ and $v = m \bmod b$, and d is a common divisor of a and b, then $u \bmod d = m \bmod d$ (see the first part: d divides a) and $v \bmod d = m \bmod d$ (for a similar reason). Therefore, $u \bmod d = v \bmod d$ in this case, as we claimed.

On the other hand, assume that $u \bmod d = v \bmod d$, i.e., $u - v$ is divisible by d. Recall the second proof of the second statement: we argued that $xa - yb = u - v$ has solutions since $\gcd)(a, b) = 1$. Now $d = \gcd(a, b)$ is no more 1, and in this case not every right hand side in a Diophantine equation is OK, it should be divisible by d. But this is exactly our assumption. Theorem is proven.

Let us look at our proof from an algorithmic viewpoint: given a, b, u, v, how can we find the required m? The first argument, with pigeons, suggests to try all m in $0 \ldots ab - 1$ until we find a good one. Not a practical thing if a and b are large (several thousand digits). But the second proof, with Diophantine equations, saves us since there exist extended Euclid algorithm that can process numbers of that size with ease.

```
from sympy import gcdex


def chinese_remainder_theorem(n1, r1, n2, r2):
    x, y, d = gcdex(n1, n2)
    assert n1 * x + n2 * y == d  # == gcd(n1, n2)
    y = -y
    assert n1 * x - n2 * y == d
    assert (r2 - r1) % d == 0
    x *= (r2-r1) // d
    y *= (r2-r1) // d
    assert n1 * x - n2 * y == r2 - r1
    return (n1 * x+r1) % (n1 * n2)


for n1, r1, n2, r2 in (
    (5, 3, 12, 7),
    (10, 3, 13, 8),
    (10, 3, 14, 1),
    (10, 3, 14, 2)
):
    result = chinese_remainder_theorem(n1, r1, n2, r2)
    print(f'If x={r1} mod {n1} and x={r2} mod {n2}, then x={result}')
```

```
If x=3 mod 5 and x=7 mod 12, then x=43
If x=3 mod 10 and x=8 mod 13, then x=73
If x=3 mod 10 and x=1 mod 14, then x=113
Traceback (most recent call last):
  File "crt4.py", line 22, in <module>
    result = chinese_remainder_theorem(n1, r1, n2, r2)
  File "crt4.py", line 9, in chinese_remainder_theorem
    assert (r2 - r1) % d == 0
AssertionError
```

In this program we do not implement ourselves the extended Euclid algorithm but import it from python modules; it returns the solution of the equation and the greatest common divisor itself. The algorithm follows our proof; the only difference is the last line that brings the answer into `[0, n1 * n2)`. Note that in python the `%` operation works correctly for all numbers, including negative, and returns the remainder in the sense of number theory (`(-1)%4` is 3, not -1).

The program finds 43 that gives remainders 3 and 7 when divided by 5 and 12. Then it finds 73 that gives remainders 3 and 8 when divided by 10 and 13. In the third call the moduli are not relatively prime but the number in question exists (see statement 3 of the theorem).

The `assert` lines show what we expect at each moment. They are checked by the python interpreter, and when we (the fourth call) ask for a number that does not exist, the corresponding line generates an error.

Problem

Assume that positive integers $n_1, \ldots, n_k$ are pairwise relatively prime, i.e., $\gcd(n_i, n_j) = 1$ for all $i \ne j$. Assume the integers r_i are given such that $0 \le r_i < n_i$. Prove that there exists some integer m such that

$$m \equiv r_1 \pmod{n_1},\ m \equiv r_2 \pmod{n_2}, \ldots,\ m \equiv r_k \pmod{n_k}.$$

Hint. This can be done step by step. First we use the Chinese remainder theorem (for two numbers) to get a number r_{12} such that $r_{12} \equiv r_1 \pmod{n_1}$ and $r_{12} \equiv r_2 \pmod{n_2}$. The first two conditions then can be replaced by one condition $m \equiv r_{12} \pmod{n_1 n_2}$. In this way the problem is reduced to a smaller instance, with $k - 1$ moduli $n_1 n2, n_3, n_4, \ldots, n_k$, and they are relatively prime: n_1 and n_2 do not share any factors with $n_3, n_4, \ldots$, so neither does their product $n_1 n_2$.

We see that an integer m can be reconstructed given $m \bmod n_1, m \bmod n_2, \ldots, m \bmod n_k$ if all n_i are pairwise relatively prime and m is not too large (say, between 0 and $n_1 n_2 \ldots n_k - 1$). This observation sometimes is used to store large numbers: instead of storing some m, we keep in the memory the list of all remainders $m \bmod n_i$. The advantage of this system (that looks quite unusual at first) is that we can multiply (and add) the numbers fast: for each modulus the operation can be performed independently, and this is easier than usual algorithm for multiplication (for addition, there is no significant difference). The catch, however, is that for this way of representing numbers it is not so easy to see which of the two given numbers is bigger (looking at the remainders only).

Completed **Go to next item**

Like **Dislike** **Report an issue**