Extended GCD Algorithm, Chinese Remainder Theorem, Fermat's Little Theorem

CS70: Discrete Mathematics and Probability Theory

UC Berkeley – Summer 2025

Lecture 8

Ref: Notes 6 and 7

Today

Extending the basic Euclid GCD algorithm
- Computing additional useful values along the way
- Using these values to find multiplicative inverses
- Other uses of Euclid: Fundamental Theorem of Arithmetic

Chinese Remainder Theorem
- Mapping from one modulus to two (or several)
- Use in speeding up computations with composite moduli

Fermat's Little Theorem
- Powers with a prime modulus
- A few tricks enabled by Fermat's Little Theorem

Euclid's GCD Algorithm – Recap

```
def euclid(x, y):
    if y == 0:
        return x

    return euclid(y, x % y)
```

Theorem: `euclid(x,y)` correctly computes $\gcd(x, y)$.

Run time: When $x \geq y$, `euclid` takes at most $2\log_2 x$ steps
$\Rightarrow$ This is linear in the *number of bits* of x
(That's fast!)

Can quickly tell if there is a multiplicative inverse for x mod m

Next Problem: So how do we compute the inverse?

Extended GCD

Euclid's Extended GCD Theorem: For any $x, y \in \mathbb{Z}$, there exist $a, b \in \mathbb{Z}$ such that $ax + by = d$ where $d = \gcd(x, y)$.

Just about existence – we'll talk about computing a and b later!

Re-stated: "We can make d out of sum of multiples of x and y."

Relation to multiplicative inverse of x modulo m?

We have $\gcd(x, m) = 1$ (otherwise no inverse!), so there are $a, b \in \mathbb{Z}$ with

$$ax + bm = 1 \qquad \Longrightarrow bm = 1 - ax \qquad \Longrightarrow ax \equiv 1 \pmod{m}$$

So a is the multiplicative inverse of $x \pmod{m}$!

Example: For $x = 12$ and $m = 35$, we have $\gcd(12, 35) = 1$, so inverse exists.

Values $a = 3$ and $b = -1$, since $3 \cdot 12 + (-1) \cdot 35 = 1$.

$\Rightarrow$ Multiplicative inverse of 12 (mod 35) is a, or 3.

Check: $3 \cdot 12 = 36$ and $36 \equiv 1 \pmod{35}$.

Pulling Multiples of x and y Out of GCD Computation

```
euclid(35,12)
  euclid(12, 11)  ;;  euclid(12, 35%12)
    euclid(11, 1)  ;;  euclid(11, 12%11)
      euclid(1,0)
        1
```

How did `euclid` get 11 from 35 and 12? $11 = 35 \bmod 12$

Another view of this operation: $35 - \lfloor \frac{35}{12} \rfloor 12 = 35 - (2)12 = 11$

How does gcd get 1 from 12 and 11?

$12 - \lfloor \frac{12}{11} \rfloor 11 = 12 - (1)11 = 1$

Algorithm finally returns 1.

But we want 1 from sum of multiples of 35 and 12?

Get 1 from 12 and 11.

$1 = 12 - (1)11 = 12 - (1)(35 - (2)12) = (3)12 + (-1)35$

Get 11 from 35 and 12 and plugin.... collect multiples of 12 and 35...

Finally: $a = 3$ and $b = -1$.

Extended GCD Algorithm

```
def extgcd(x, y):
    if y == 0:
        return (x, 1, 0)

    (d, a, b) = extgcd(y, x % y)
    return (d, b, a - b*(x // y)) # Note: // is integer division
```

Claim: Returns (d, a, b): $d = gcd(x, y)$ and $d = ax + by$.

Example:

```
    extgcd(35,12)
      extgcd(12, 11)
        extgcd(11, 1)
          extgcd(1,0)
          return (1,1,0) ;; 1 = (1)1 + (0) 0
        return (1,0,1)   ;; 1 = (0)11 + (1)1
      return (1,1,-1)    ;; 1 = (1)12 + (-1)11
   return (1,-1, 3)      ;; 1 = (-1)35 +(3)12
```

Extended GCD Algorithm: Correctness

```
def extgcd(x, y):
    if y == 0:
        return (x, 1, 0)

    (d, a, b) = extgcd(y, x % y)
    return (d, b, a - b*(x // y)) # Note: // is integer division
```

Theorem: `extgcd(x,y` returns (d, a, b), where $d = \gcd(a, b)$ and $d = ax + by$.

Proof: Computation of d is exactly as before, so $d = \gcd(a, b)$. We prove the remaining property by (strong) induction on y.

Base case ($y = 0$)*:* `extgcd(x,0)` returns (x,1,0), we know $x = d$ and $1 \cdot x + 0 \cdot 0 = x$ ✓

Extended GCD Algorithm: Correctness continued

Induction Hypothesis: Assume that for all $x' \geq y'$ and $y' < y$, $\texttt{extgcd}(x', y')$ returns (d, a, b) with $d = a \cdot x' + b \cdot y'$.

Induction Step: We prove that at y, `extgcd(x,y)` returns (d, A, B) with $d = A \cdot x + B \cdot y$.

Makes a recursive call for $\texttt{extgcd}(y, x \bmod y)$. Since $(x \bmod y) < y$ the induction hypothesis states that this returns (d, a, b) with $a \cdot y + b \cdot (x \bmod y) = d$.

Given this value from the recursive call, `extgcd` returns (d, A, B) calculated as $A = b$ and $B = a - b \cdot \lfloor \frac{x}{y} \rfloor$ (from the algorithm).

$$\begin{aligned}
A \cdot x + B \cdot y &= b \cdot x + (a - b \cdot \lfloor \tfrac{x}{y} \rfloor) y \\
&= b \cdot x + a \cdot y - b \lfloor \tfrac{x}{y} \rfloor y \\
&= a \cdot y + b \cdot (x - \lfloor \tfrac{x}{y} \rfloor y) \\
&= a \cdot y + b \cdot (x \bmod y)
\end{aligned}$$

This last formula matches the induction hypothesis, so is equal to d. $\square$

Non-Recursive Hand Calculation Method

Example for 7 and 60 — note $\gcd(7,60) = 1$

$$\begin{array}{lclll} 7(0) + 60(1) & = & 60 & (1) \\ 7(1) + 60(0) & = & 7 & (2) \end{array}$$

Idea: subtract largest multiple of the second one you can keeping RHS smaller

That multiple is $\lfloor \frac{60}{7} \rfloor = 8$

$$\begin{array}{rlcll} & 7(0) + 60(1) & = & 60 & (1) \\ - & 7(8) + 60(0) & = & 56 & (2 \text{ multiple}) \\ \hline & 7(-8) + 60(1) & = & 4 & (3) \end{array}$$

Do it again with (2) and (3) [multiple is $\lfloor \frac{7}{4} \rfloor = 1$

$$\begin{array}{rlcll} & 7(1) + 60(0) & = & 7 & (2) \\ - & 7(-8) + 60(1) & = & 4 & (3 \text{ multiple}) \\ \hline & 7(9) + 60(-1) & = & 3 & (4) \end{array}$$

And again....

$$\begin{array}{rlcll} & 7(-8) + 60(1) & = & 4 & (3) \\ - & 7(9) + 60(-1) & = & 3 & (4) \\ \hline & \color{red}{7(-17) + 60(2)} & \color{red}{=} & \color{red}{1} & \end{array}$$

Multiplicative inverse of 7 $\pmod{60}$ is $-17 \equiv 43 \pmod{60}$

Wrap-up of Computing Multiplicative Inverses

Conclusion: Can find multiplicative inverses with n-bit modulus in $O(n)$ time!

Very different from grade school: try 1, try 2, try 3... optimized: $2^{n/2}$ time.

Inverse of $500,000,357$ modulo $1,000,000,000,000$?
≤ 80 divisions.
versus $1,000,000$

Soon we'll see cryptography that uses *very* large numbers
Example: Numbers with 1024 bits
Euclid: At most 2048 divisions to find multiplicative inverse
Grade School: 1000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000 divisions

This kind of cryptography is impossible without an algorithm like Euclid's.

Fundamental Theorem of Arithmetic

Euclid's Extended GCD Theorem is useful for things beyond computation.

Theorem: Every natural number can be written as the product of primes.

Proof: Uses strong induction – existence of product of primes:
Case 1: n is prime. Done.
Case 2: n is not prime, so can be written as $n = a \cdot b$. By IH, both a and b can be written as the product of primes. $\square$

Theorem: The prime factorization of n is unique up to reordering.

Proof idea: We use Euclid's Extended GCD Theorem!

Fundamental Theorem of Arithmetic: Every natural number can be written as a unique (up to reordering) product of primes.

Euclid For Proofs About Shared Factors

Claim: For $x, y, z \in \mathbb{Z}^+$ with $gcd(x, y) = 1$ and $x \mid yz$ then $x \mid z$.

Idea (restatement): x doesn't share factors with y so it must divide z.

Euclid: There exists $a, b \in \mathbb{Z}$ such that $1 = ax + by \implies z = axz + byz$.

Observe: $x \mid axz$ (obviously) and $x \mid byz$ (since $x \mid yz$), and x divides the sum.
$\implies x \mid axz + byz$, and since $axz + byz = z$ we have $x \mid z$. $\square$

So to prove Fundamental Theorem of Arithmetic:

- Proof by contradiction: Assume two factorizations $p_1 \cdots p_k$ and $q_1 \cdots q_\ell$
- Induction: p_1 divides both (same number).
  - Using claim: p_1 divides $q_1 \cdot q_{\ell-1}$ or q_ℓ.
- Conclusion: $p_1 = q_i$ for some i.

Values Modulo Product of Two Primes

x	$x \bmod 3$	$x \bmod 5$
0	0	0
1	1	1
2	2	2
3	0	3
4	1	4
5	2	0
6	0	1
7	1	2
8	2	3
9	0	4
10	1	0
11	2	1
12	0	2
13	1	3
14	2	4

Table shows x from 0 to 14 – so $x \pmod{15}$

Any x with $x \equiv 1 \pmod 3$ and $x \equiv 4 \pmod 5$?
Yes! $x = 4$

Any x with $x \equiv 2 \pmod 3$ and $x \equiv 3 \pmod 5$?
Yes! $x = 8$

x any a, b: $x \equiv a \pmod 3$ and $x \equiv b \pmod 5$?
Yes! Check all – or prove a general theorem!

Chinese Remainder Theorem (2 modulus version)

Theorem: For m, n with $\gcd(m,n) = 1$, and any a, b, there is exactly one $x \in \{0, 1, \ldots, mn-1\}$ with $x \equiv a \pmod{m}$ and $x \equiv b \pmod{n}$.

Note: Previous table had $m = 3$, $n = 5$, two primes. The requirement isn't so strict: m and n only need to be relatively prime. (*Example on next slide...*)

Proof: First consider existence of a solution.

$\gcd(n,m) = 1$ so compute $s = n^{-1} \pmod{m}$, and consider integer $u = s \cdot n$

$u \bmod m = 1$ $\quad$ $u \bmod n = 0$

Similarly, compute $t = m^{-1} \pmod{n}$, and consider $v = t \cdot m$

$v \bmod n = 1$ $\quad$ $v \bmod m = 0$

Now compute $x = (a \cdot u + b \cdot v) \bmod mn$

Consider mod m: $\quad x \equiv a \cdot u + b \cdot v \equiv a \cdot 1 + b \cdot 0 \equiv a \pmod{m}$

Consider mod n: $\quad x \equiv a \cdot u + b \cdot v \equiv a \cdot 0 + b \cdot 1 \equiv b \pmod{n}$

Unique: For any $x \in \{0, 1, \ldots, mn-1\}$ compute $a = x \bmod m$ and $b = x \bmod n$

Can map $x \mapsto (a,b)$ and $(a,b) \mapsto x$

$\Rightarrow$ Mapping is a bijection (one-to-one) so solution is unique. $\square$

Using the Chinese Remainder Theorem

Proof that solution x exists was constructive, so can use it as to compute

Ex: Let's find $x \pmod{1155}$ with $x \equiv 17 \pmod{33}$ and $x \equiv 14 \pmod{35}$

So $n = 33$, $m = 35$, $nm = 1155$, $a = 17$, and $b = 14$

$\Rightarrow$ Note! n and m are *not prime* – but are *relatively prime*!

We typically use prime moduli, but this is not required!

Compute $s = n^{-1} \pmod{m} = 33^{-1} \pmod{35}$ This is 17

Computed using `extgcd`: Check $33 \cdot 17 = 561 = 16 \cdot 35 + 1$

$u = s \cdot n = 17 \cdot 35 = 595$

Compute $t = m^{-1} \pmod{n} = 35^{-1} \pmod{33}$ This is 17 (coincidence!)

$v = t \cdot m = 17 \cdot 33 = 561$

Finally, compute $a \cdot u + b \cdot v = 17 \cdot 595 + 14 \cdot 561 = 17696$

Then reduce: $x = 17969 \bmod 1155 = 644$

Did it really work?

$644 \bmod 33 = 17$ (since $644 = 19 \cdot 33 + 17$)

$644 \bmod 35 = 14$ (since $644 = 18 \cdot 35 + 14$)

Chinese Remainder Theorem: Extension and Uses

Extension

No need to restrict to just two moduli

Use $m_1, m_2, \ldots, m_k$ that have $\gcd(m_i, m_j) = 1$ for all $i \neq j$ (pairwise co-prime)

Let $m = m_1 \cdot m_2 \cdots m_k$

Given values $x_1, x_2, \ldots$ ...

... a unique solution $x \pmod{m}$ such that $x_1 \equiv x \bmod m_1$, $x_2 \equiv x \bmod m_2$, ...

A Practical Use

For input x, we want to do some long computation $f(x) \bmod mn$ (e.g, powering)

Instead:

1. Compute $x_m = x \bmod m$
2. Compute $x_n = x \bmod n$
3. Compute $y_m = f(x_m) \bmod m$
4. Compute $y_n = f(x_n) \bmod n$
5. Combine results y_m and y_n using CRT to find result $y \pmod{mn}$

Steps 3 and 4 work on smaller numbers, so can be faster overall

If steps 3 and 4 an be done *in parallel* can be *much* faster!

Hardware accelerators for cryptography use this!

Playing with Numbers... Just Because...

Recall proof that $\gcd(x,m)=1 \implies x$ has a mult inverse mod m
$\Rightarrow$ Looked at products $0x, 1x, \ldots, (m-1)x$ (all mod m)

Showed that products contain exactly one copy of every value $0, 1, \ldots, m-1$

Remember Steve's advice? Be exploratory. Be playful.
What else can we do with these products?
What if we multiplied all the non-zero values together? Why? Why not?

Products just rearrange all values, so equal to product of all values...

$$1x \cdot 2x \cdot \cdots \cdot (m-1)x \equiv 1 \cdot 2 \cdot \cdots \cdot (m-1) \pmod{m}$$
$$(1 \cdot 2 \cdot \cdots \cdot (m-1))x^{m-1} \equiv 1 \cdot 2 \cdot \cdots \cdot (m-1) \pmod{m}$$

Wouldn't it be cool if we could cancel out $1 \cdot 2 \cdot \cdots \cdot (m-1)$ from both sides?
To do that, need a multiplicative inverse or $\gcd(1 \cdot 2 \cdot \cdots \cdot (m-1), m) = 1$
True if and only if m is prime – this seems important...

Congratulations! By being playful, you are as good a mathematician as Fermat!
If only it were really that easy....

Fermat's Little Theorem

Fermat's Little† **Theorem:** For prime p, and $a \not\equiv 0 \pmod{p}$, $a^{p-1} \equiv 1 \pmod{p}$.

Proof: Consider $S = \{a \cdot 1, \ldots, a \cdot (p-1)\}$. All different modulo p since a has an inverse modulo p (so multiplying by a is a bijection). Therefore

$$(a \cdot 1) \cdot (a \cdot 2) \cdots (a \cdot (p-1)) \equiv 1 \cdot 2 \cdots (p-1) \pmod{p},$$

since multiplication is commutative.

$$a^{(p-1)}(1 \cdots (p-1)) \equiv (1 \cdots (p-1)) \pmod{p}.$$

Since p is prime, its smallest factor > 1 is p, and so $1 \cdots (p-1)$ is relatively prime to p and hence has a multiplicative inverse. Multiply each side above by this multiplicative inverse to get

$$a^{(p-1)} \equiv 1 \pmod{p}. \qquad \square$$

† *Not Fermat's Last Theorem. Yes, both "FLT." Yes, can be confusing.*

Proof Illustration with Numbers

We'll use $p = 5$ and $a = 2$

First sequence: $1, 2, 3, 4$

Second sequence: $(2 \cdot 1), (2 \cdot 2), (2 \cdot 3), (2 \cdot 4) \;=\; 2, 4, 1, 3 \pmod 5$.

Multiply LHS and simplify: $(2 \cdot 1) \cdot (2 \cdot 2) \cdot (2 \cdot 3) \cdot (2 \cdot 4) = 2^4 (1 \cdot 2 \cdot 3 \cdot 4)$

Multiply RHS and reorder: $2 \cdot 4 \cdot 1 \cdot 3 = 1 \cdot 2 \cdot 3 \cdot 4$
Because multiplication is commutative

Was the *same* sequence mod 5, so $2^4 \cdot 1 \cdot 2 \cdot 3 \cdot 4 \equiv 1 \cdot 2 \cdot 3 \cdot 4 \pmod 5$

Since 5 is prime, no shared factors with any of 1, 2, 3, or 4
$\Rightarrow \gcd(1 \cdot 2 \cdot 3 \cdot 4, 5) = 1$
$\Rightarrow 1 \cdot 2 \cdot 3 \cdot 4$ has a mult inverse mod 5, so can cancel out

Therefore, $2^4 \equiv 1 \pmod 5$

Really? $2^4 = 16$ and $16 \bmod 5 = 1$ – so yes, really.

Concept Check!

Question: Which of the following was used in Fermat's theorem proof?

(A) The mapping $f(x) = ax \mod p$ is a bijection.

(B) Multiplying a number by 1, gives the number.

(C) When p is prime, $\gcd(p,(p-1)!) = 1$

(D) Multiplying a number by 0 gives 0.

(E) Multiplying elements of sets A and B together is the same if $A = B$.

Answer: (A), (C), and (E)

Fermat's Little Theorem Tricks

FLT: For prime p, and $a \not\equiv 0 \pmod{p}$, $a^{p-1} \equiv 1 \pmod{p}$.

Trick #1: Simplifying powering by reducing the exponent.

What is $2^{101} \pmod{7}$?

What is quotient and remainder dividing exponent (101) by $p-1$ (6)?

$101 = 6 \cdot 16 + 5$, so $2^{101} \equiv 2^{6 \cdot 16 + 5} \equiv \left(2^6\right)^{16} \cdot 2^5 \equiv 2^5 \equiv 32 \pmod{7}$

$32 \bmod 7 = 4$, so $2^{101} \equiv 4 \pmod{7}$

A bit easier than using $2^{101} = 2535301200456458802993406410752$

Trick #2: Computing multiplicative inverses mod a prime p.

Note that $a^{p-1} \equiv a \cdot a^{p-2} \equiv 1 \pmod{p}$

$\Rightarrow$ so $a^{p-2} \bmod p$ is the multiplicative inverse of a

Example: Multiplicative inverse of $4 \pmod{7}$?

$$4^5 = 1024 \text{ and } 1024 \bmod 7 = 2$$

Using Python: "`p=7; pow(4,p-2,p)`" gives `2`.

Fermat's Little Theorem Almost-Tricks

FLT: For prime p, and $a \not\equiv 0 \pmod{p}$, $a^{p-1} \equiv 1 \pmod{p}$.

Trick #3: Almost.... Can we use FLT to *test* for primality?

Example: Is 5153642624137 prime?

Could try dividing things into it... slow.

Or:

```
>>> n=5153642624137
>>> pow(5, n-1, n)
15625
```

So $a^{n-1} \not\equiv 1 \pmod{n}$: n doesn't satisfy property all primes must

So... n is not prime

Correct in this case, but will this always work? No - two problems:

1. For all composite n, *some* choices of a will give 1
   Solution: Usually... Less than half of a's, so pick at random (and repeat!)
2. For *some* n, formula holds for all a's (Carmichael numbers)
   Solution: A bit harder, but can solve....

Result: Miller-Rabin primality testing algorithm

Berkeley connection! Based on Gary Miller's Ph.D. dissertation from Berkeley.

Summary

Extended Euclid: Find a, b where $ax + by = gcd(x, y)$

Idea: compute a, b recursively (euclid), or iteratively

Inverse: $ax + by \equiv ax \equiv gcd(x, y) \pmod{y}$

If $gcd(x, y) = 1$, we have $ax \equiv 1 \pmod{y}$

$\longrightarrow a \equiv x^{-1} \pmod{y}$

Fundamental Theorem of Arithmetic: Unique prime factorization of any n

Claim: if $p|n$ and $n = xy$, $p|x$ of $p|x$.

Proof relies on Extended Euclid GCD Theorem

Fundamental Theorem follows using induction + contradiction. Chinese

Remainder Theorem:

If $gcd(n, m) = 1$ then $x = a \pmod{n}$, $x = b \pmod{m}$ unique sol.

Proof: Find $u = 1 \pmod{n}$, $u = 0 \pmod{m}$,

and $v = 0 \pmod{n}$, $v = 1 \pmod{m}$.

Then: $x = au + bv = a \pmod{n}$

Fermat: For prime p, $a^{p-1} \equiv 1 \pmod{p}$

Proof Idea: $f(x) = a \cdot x \pmod{p}$ is bijection on $S = \{1, \ldots, p-1\}$.

Multiply domain elts and range elts – cancel and left with just a^{p-1} in range