Ch 3: Algorithms & Complexity

3.2: Growth of Functions (Big-O)

Definition 1. Let f, g be functions $\mathbb{N} \to \mathbb{R}$ or $\mathbb{R} \to \mathbb{R}$. We say $\mathbf{f(x) = O(g(x))}$ (Big-O) if $\exists C, k$ (witnesses) s.t.

$$|f(x)| \leq C|g(x)| \quad \text{for all } x > k.$$

Definition 2. $\mathbf{f(x) = \Omega(g(x))}$ (Big-Omega) if $\exists C, k$ (positive witnesses) s.t.

$$|f(x)| \geq C|g(x)| \quad \text{for all } x > k.$$

Definition 3. $\mathbf{f(x) = \Theta(g(x))}$ (Big-Theta) if $f(x) = O(g(x))$ and $f(x) = \Omega(g(x))$. This is equivalent to $\exists C_1, C_2, k$ s.t.

$$C_1|g(x)| \leq |f(x)| \leq C_2|g(x)| \quad \text{for all } x > k.$$

Common Growth Functions (slowest to fastest): $O(1) < O(\log n) < O(n) < O(n \log n) < O(n^2) < O(n^c) < O(b^n) < O(n!)$

Properties:

- $f_1 = O(g_1), f_2 = O(g_2) \implies f_1 + f_2 = O(\max(g_1, g_2))$
- $f_1 = O(g_1), f_2 = O(g_2) \implies f_1 f_2 = O(g_1 g_2)$

Ch 4.1-4.3: Number Theory

4.1: Divisibility

Definition 4. If $a, b \in \mathbb{Z}$ with $a \neq 0$, we say **a** divides **b** (written $a|b$) if $\exists c \in \mathbb{Z}$ s.t. $b = ac$.

Property 1 (Properties of Divisibility)**.**

- If $a|b$ and $a|c$, then $a|(mb + nc)$ for any $m, n \in \mathbb{Z}$.
- If $a|b$ and $b|c$, then $a|c$.

Theorem 1 (Division Algorithm)**.** Let $a \in \mathbb{Z}$ and $d \in \mathbb{Z}^+$. Then $\exists!$ integers q (quotient) and r (remainder) s.t. $a = dq + r$ and $0 \leq r < d$.

4.1: Modular Arithmetic

Definition 5. Let $m \in \mathbb{Z}^+$. We say $\mathbf{a \equiv b \pmod{m}}$ (a is congruent to b modulo m) if $m|(a - b)$.

Equivalent statements:

- $a \equiv b \pmod{m}$
- $m|(a - b)$
- $a = b + km$ for some $k \in \mathbb{Z}$
- $a \pmod{m} = b \pmod{m}$

Modular Arithmetic Properties:

- $(a + b) \pmod{m} = ((a \pmod{m}) + (b \pmod{m})) \pmod{m}$
- $(a \cdot b) \pmod{m} = ((a \pmod{m}) \cdot (b \pmod{m})) \pmod{m}$

4.2: Integer Representation

For an n-bit number:

- **Sign Bit:** MSB (left-most) is 0 for positive, 1 for negative.
- **One's Complement:** To negate, flip all bits. (e.g., $5 = 00000101$, $-5 = 11111010$).
- **Two's Complement:** Standard method. To negate: take the positive n-bit represenation, find the one's complement (flip all bits), and add 1. (e.g., $5 = 00000101$, $-5 = 11111011$)

4.2: Primes

Definition 6. A prime $p > 1$ is an integer whose only positive divisors are 1 and p. A number $n > 1$ that is not prime is **composite**.

Theorem 2 (Fundamental Theorem of Arithmetic)**.** Every integer $n > 1$ can be written uniquely as a prime or as a product of two or more primes in non-decreasing order. $n = p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k}$

Theorem 3. If n is composite, it has a prime factor $\leq \sqrt{n}$.

4.3: GCD & LCM

Definition 7. $\gcd(\mathbf{a}, \mathbf{b})$: The largest integer d s.t. $d|a$ and $d|b$. If $\gcd(a, b) = 1$, a and b are **relatively prime**.

Definition 8. $\text{lcm}(\mathbf{a}, \mathbf{b})$: The smallest integer $m > 0$ s.t. $a|m$ and $b|m$.

Prime Factorization Method: $a = p_1^{a_1} \cdots p_n^{a_n}$, $b = p_1^{b_1} \cdots p_n^{b_n}$

- $\gcd(a, b) = p_1^{\min(a_1, b_1)} \cdots p_n^{\min(a_n, b_n)}$
- $\text{lcm}(a, b) = p_1^{\max(a_1, b_1)} \cdots p_n^{\max(a_n, b_n)}$

Theorem 4. For $a, b \in \mathbb{Z}^+$, $a \cdot b = \gcd(a, b) \cdot \text{lcm}(a, b)$

Euclidean Algorithm: Finds $\gcd(a, b)$ for $a \geq b > 0$. Let $r_0 = a$, $r_1 = b$.

$$\begin{aligned}
r_0 &= r_1 q_1 + r_2 \quad (0 \leq r_2 < r_1) \\
r_1 &= r_2 q_2 + r_3 \quad (0 \leq r_3 < r_2) \\
&\vdots \\
r_{n-2} &= r_{n-1} q_{n-1} + r_n \quad (0 \leq r_n < r_{n-1}) \\
r_{n-1} &= r_n q_n + 0
\end{aligned}$$

$\gcd(a, b) = r_n$ (the last non-zero remainder).

Ch 4.6: Cryptography

Modular Inverse

Definition 9. An integer a^{-1} is a **modular inverse** of a modulo m if $a \cdot a^{-1} \equiv 1 \pmod{m}$. It exists $\iff \gcd(a, m) = 1$.

Decryption Functions (Affine/Shift)

If Encryption is $C = E(P) = (aP + b) \pmod{m}$:

1. To decrypt, you need the inverse function $P = D(C)$.
2. Algebraically solve for P:
$$C - b \equiv aP \pmod{m}$$
3. Multiply both sides by a^{-1}:
$$a^{-1}(C - b) \equiv P \pmod{m}$$
4. **Decryption Function:** $D(C) = a^{-1}(C - b) \pmod{m}$.

Transposition Decryption Example

Decrypt **"ATNACDXTAWTKA"** w/ key **"4, 1, 3, 2"**.

1. **Find Dims:** $N = 13, k = 4$. Rows $R = \lceil 13/4 \rceil = 4$.
2. **Col Lengths:** $13 \pmod 4 = 1$. The first **1** column in the *sorted* key order gets an extra char.
   - Key '1' $\to$ 4 chars.
   - Key '2' $\to$ 3 chars.
   - Key '3' $\to$ 3 chars.
   - Key '4' $\to$ 3 chars.
3. **Break Ciphertext:** Break $N = 13$ string by key order: (Key 4) (Key 1) (Key 3) (Key 2).
   - Key 4 (len 3): "ATN"
   - Key 1 (len 4): "ACDX"
   - Key 3 (len 3): "TAW"
   - Key 2 (len 3): "TKA"
4. **Fill Grid & Read Rows:** Write pieces into columns **by key number**.

Col 1 (Key 1)	**Col 2** (Key 2)	**Col 3** (Key 3)	**Col 4** (Key 4)
A	T	T	A
C	K	A	T
D	A	W	N
X			

Read rows: "ATTACKATDAWNX"

Fast Modular Exponentiation

To compute $b^n \pmod m$, write n in binary. Compute $b, b^2, b^4, \ldots \pmod m$ by repeated squaring. Multiply terms where $a_i = 1$.

Example 1. $7^{11} \pmod{13}$: $11 = (1011)_2 = 8 + 2 + 1$. $7^1 \equiv 7, 7^2 \equiv 10, 7^4 \equiv 9, 7^8 \equiv 3 \pmod{13}$. $7^{11} = 7^8 \cdot 7^2 \cdot 7^1 \equiv 3 \cdot 10 \cdot 7 \equiv 2 \pmod{13}$.

Ch 5: Induction

5.1: Mathematical Induction

Used to prove $P(n)$ for all integers $n \geq n_0$.

Template (Weak Induction):

1. **Basis Step:** Show $P(n_0)$ is true.
2. **Inductive Hypothesis (IH):** Assume $P(k)$ is true for arbitrary $k \geq n_0$.
3. **Inductive Step:** Show $P(k) \implies P(k+1)$.
4. **Conclusion:** By induction, $P(n)$ is true $\forall n \geq n_0$.

Example 2 (Summation)**.** Prove $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$ for $n \geq 1$.

1. **Basis ($n = 1$):** $1 = \frac{1(2)}{2} = 1$. True.
2. **IH:** Assume $\sum_{i=1}^{k} i = \frac{k(k+1)}{2}$ for $k \geq 1$.
3. **Step:** $\sum_{i=1}^{k+1} i = \left(\sum_{i=1}^{k} i\right) + (k+1) = \frac{k(k+1)}{2} + (k+1) = (k+1)(\frac{k}{2} + 1) = \frac{(k+1)(k+2)}{2}$.

5.2: Strong Induction

Template (Strong Induction):

1. **Basis Step(s):** Show $P(n_0)$ (and potentially $P(n_0 + 1)$ etc.) is true.
2. **IH:** Assume $P(j)$ is true for **all** j s.t. $n_0 \leq j \leq k$.
3. **Inductive Step:** Show $[P(n_0) \wedge \cdots \wedge P(k)] \implies P(k+1)$.

Theorem 5 (Well-Ordering Principle)**.** Every non-empty set of non-negative integers has a least element.