0.1 Chapter 0.

- Asymptotic notation: big $O$, $\Theta$, big $\Omega$, little $o$, little $\omega$.

0.2 Chapter 1.

- Modular arithmetic. Can use equivalents in calculations for plus and times. Prove this for times: $a = x \mod n$, $b = y \mod n$. That is $a = x + in$ and $b = y + jn$, thus $ab = xy + xjn + yin + ijn^2 = xy + n(xj + yin + ijn)$, or $ab = xy \mod n$.

- Modulo exponentiation: $\text{modexp}(a, b) = x$.
  Return $x^2 \equiv x^{b/2}$.

- Modulo inverses from Euclid’s theorem: for $(a, b)$ there exist integer $i$, $j$ such that $\gcd(a, b) = ax + by$.
  Runtime for Euclid’s? (Basic idea: $\gcd(a, b) = \gcd(a, a \mod b)$ Use formula recursively.)

- Show that every $a$ not divisible by $p$ has an inverse mod prime $p$.

- Fermat’s little theorem. Can use equivalents in exponents. For a prime, $a^{p-1} \mod p = 1$. Proof: $ai \neq aj \mod p$ unless $j = i$. Thus, $a^{p-1}(p-1)! = (p-1)! \mod p$.

- Prime number theorem: random number with $b$ bits has probability $\Omega(1/b)$ of being prime. Probability that $x$ is prime is approximately $1/\ln x$.

- RSA: Private-key: $(N = pq, e)$. Public-key $(N, d = e^{-1} \mod (p-1)(q-1))$. Encode $m$: $m^e \mod N$.
  Decode $x$: $x^d \mod N$.

- RSA decoding gives back the original message since $a^{(p-1)(q-1)} = 1 \mod N$ for $a$ not divisible by $p$ or $q$. Euler’s theorem.

- Hashing: universal hash function $h() \rightarrow [1, n]$: for all $x, y$, $\Pr_{h \in H}[h(x) = h(y)] = 1/n$. Example.

0.3 Chapter 2

- Karatsuba’s algorithm:

  $$(a_{\text{top}}2^{n/2} + a_{\text{bot}})(b_{\text{top}}2^{n/2} + b_{\text{bot}}) = 2^n a_{\text{top}}b_{\text{top}}op + (a_{\text{top}}b_{\text{bot}} + a_{\text{bot}}b_{\text{top}})2^{n/2} + a_{\text{bot}}b_{\text{bot}}.$$ 

  Four multiplies of $n/2$ bit words. $T(n) = 4T(n/2) + n = O(n^2)$. 

0-1
Lecture 0 Midterm I Review: 9.30.08

• But, \((a_{\text{top}} + a_{\text{bot}})(b_{\text{bot}} + b_{\text{top}}) = a_{\text{top}}b_{\text{top}} + b_{\text{bot}}a_{\text{bot}} + (a_{\text{top}}b_{\text{bot}} + a_{\text{bot}}b_{\text{top}})\). The paranthesized quantity is the middle quantity. Thus, the recurrence is \(T(n) = 3T(n/2) + n = O(n^{\log_2 3})\).

• Master’s theorem: \(T(n) = aT(n/b) + n^c\). If \(c > \log_b a\), \(T(n) = O(n^c)\), if \(c = \log_b a\), \(T(n) = O(n^c \log n)\), and if \(c < \log_b a\) then \(T(n) = n^{\log_b a}\).

• Recursion tree. Number of leaves is \(O(n^{\log_b a})\). Depth is \(\log_b n\). Cost at root is \(n^c\). Cases are root dominates, all equal, leaves dominate.

• Mergesort. Sorting requires \(\Omega(n \log n)\) time.

• Matrix multiply. Can do this with 7 submatrix multiplies. Thus, recursion is \(T(n) = 7T(n/2) + O(n^2)\). This is \(O(n^{\log_2 7})\). Less than \(O(n^3)\).

• FFT. What are the \(n\) roots of unity. Definition of the FFT: \(FFT(a_1, \ldots, a_n) = A(\omega^i)\). For multiplying polynomials: \(FFT^{-1}(FFT(a) \cdot FFT(b))\).

• Algorithm: Take FFT of odd and even. Then combine. Each subproblem used for two outputs. Recurse. FFT procedure. \(T(n) = 2T(n/2) + O(n)\).

0.4 Chapter 3

• Matrix representation: fast lookup? \(O(n^2)\) space. Graph. Adjacency list representation: are \((i, j)\) connected? Space? Get a list of neighbors?

• Depth first search.


• Directed graphs. Pre, post ordering. \((u, v)\) is back edge \([\text{pre}[u], \text{post}[u]]\) contained in \([\text{pre}[v], \text{post}[v]]\). The other way, it is forward or tree edge. If non-intersecting: cross edge, and \(\text{post}[v]\) is before \(\text{pre}[u]\).

• Topological sort of dag. Two algorithms: Repeatedly find source. or Reverse post ordering number: since for edge \((u, v)\), \(\text{post}[u] \geq \text{post}[v]\). Why does this work?

• Strongly connected. Path from \(i\) to \(j\) and \(j\) to \(i\). Partitions graph: if \(i\) in differenct components, then the union is strongly connected. DAG metagraph of components.

• Algorithm: dfs of reverse graph. Dfs ordering by reverse post ordering number: Each explore finds current sink component.