ACM Lab – HS 2015

Prof. Dr. Angelika Steger

September 24, 2015

Exercise: Primes

- Task: output all primes in some interval $\{m \ldots, n\}$
- Slow solution (20): for each k try all possible divisors
- Medium solution (30): for each k try all possible divisors up to $\sqrt{k}$
- Fast solution (50): sieve of eratosthenes

Exercise: Primes

- Task: output all primes in some interval $\{m \ldots, n\}$
- Slow solution (20): for each k try all possible divisors
- Medium solution (30): for each k try all possible divisors up to $\sqrt{k}$
- Fast solution (50): sieve of eratosthenes

Exercise: Primes

- Task: output all primes in some interval $\{m \ldots, n\}$
- Slow solution (20): for each k try all possible divisors
- Medium solution (30): for each k try all possible divisors up to $\sqrt{k}$
- Fast solution (50): sieve of eratosthenes

Exercise: Primes

- Task: output all primes in some interval $\{m \ldots, n\}$
- Slow solution (20): for each k try all possible divisors
- Medium solution (30): for each k try all possible divisors up to $\sqrt{k}$
- Fast solution (50): sieve of eratosthenes

Exercise: Even Pairs

- Task: count number of even interval sums of binary sequence $S = S_1, \ldots, S_n$
- Slow solution (30): try all intervals
- Medium solution (30): try all intervals with partial sums trick
- Fast solution (50): two partial sums contribute to an even interval sum if either both are even or both are odd

Exercise: Even Pairs

- Task: count number of even interval sums of binary sequence $S = S_1, \ldots, S_n$
- Slow solution (30): try all intervals
- Medium solution (30): try all intervals with partial sums trick
- Fast solution (50): two partial sums contribute to an even interval sum if either both are even or both are odd

Exercise: Even Pairs

- Task: count number of even interval sums of binary sequence $S = S_1, \ldots, S_n$
- Slow solution (30): try all intervals
- Medium solution (30): try all intervals with partial sums trick
- Fast solution (50): two partial sums contribute to an even interval sum if either both are even or both are odd

Exercise: Even Pairs

- Task: count number of even interval sums of binary sequence $S = S_1, \ldots, S_n$
- Slow solution (30): try all intervals
- Medium solution (30): try all intervals with partial sums trick
- Fast solution (50): two partial sums contribute to an even interval sum if either both are even or both are odd

Running Time: Rule of Thumb

- Rule of Thumb: Processor can do 1M operations per second, Timelimit is 3 seconds.
- $n \approx 10^6$: Algorithm should be $\mathcal{O}(n)$
- $n \approx 10^5$: Algorithm should be $\mathcal{O}(n \log n)$
- $n \approx 10^3$: Algorithm should be $\mathcal{O}(n^2)$
- $n \approx 10^2$: Algorithm should be $\mathcal{O}(n^3)$
- $n \approx 50$: Algorithm should be $\mathcal{O}(n^4)$
- $n \approx 20$: Algorithm should be $\mathcal{O}(n^5)$ or $\mathcal{O}(2^n)$
- $n \approx 10$: Algorithm should be $\mathcal{O}(n^6)$ or $\mathcal{O}(n!)$

Running Time: Rule of Thumb

- Rule of Thumb: Processor can do 1M operations per second, Timelimit is 3 seconds.
- $n \approx 10^6$: Algorithm should be $\mathcal{O}(n)$
- $n \approx 10^5$: Algorithm should be $\mathcal{O}(n \log n)$
- $n \approx 10^3$: Algorithm should be $\mathcal{O}(n^2)$
- $n \approx 10^2$: Algorithm should be $\mathcal{O}(n^3)$
- $n \approx 50$: Algorithm should be $\mathcal{O}(n^4)$
- $n \approx 20$: Algorithm should be $\mathcal{O}(n^5)$ or $\mathcal{O}(2^n)$
- $n \approx 10$: Algorithm should be $\mathcal{O}(n^6)$ or $\mathcal{O}(n!)$

Running Time: Rule of Thumb

- Rule of Thumb: Processor can do 1M operations per second, Timelimit is 3 seconds.
- $n \approx 10^6$: Algorithm should be $\mathcal{O}(n)$
- $n \approx 10^5$: Algorithm should be $\mathcal{O}(n \log n)$
- $n \approx 10^3$: Algorithm should be $\mathcal{O}(n^2)$
- $n \approx 10^2$: Algorithm should be $\mathcal{O}(n^3)$
- $n \approx 50$: Algorithm should be $\mathcal{O}(n^4)$
- $n \approx 20$: Algorithm should be $\mathcal{O}(n^5)$ or $\mathcal{O}(2^n)$
- $n \approx 10$: Algorithm should be $\mathcal{O}(n^6)$ or $\mathcal{O}(n!)$

Running Time: Rule of Thumb

- Rule of Thumb: Processor can do 1M operations per second, Timelimit is 3 seconds.
- $n \approx 10^6$: Algorithm should be $\mathcal{O}(n)$
- $n \approx 10^5$: Algorithm should be $\mathcal{O}(n \log n)$
- $n \approx 10^3$: Algorithm should be $\mathcal{O}(n^2)$
- $n \approx 10^2$: Algorithm should be $\mathcal{O}(n^3)$
- $n \approx 50$: Algorithm should be $\mathcal{O}(n^4)$
- $n \approx 20$: Algorithm should be $\mathcal{O}(n^5)$ or $\mathcal{O}(2^n)$
- $n \approx 10$: Algorithm should be $\mathcal{O}(n^6)$ or $\mathcal{O}(n!)$

Running Time: Rule of Thumb

- Rule of Thumb: Processor can do 1M operations per second, Timelimit is 3 seconds.
- $n \approx 10^6$: Algorithm should be $\mathcal{O}(n)$
- $n \approx 10^5$: Algorithm should be $\mathcal{O}(n \log n)$
- $n \approx 10^3$: Algorithm should be $\mathcal{O}(n^2)$
- $n \approx 10^2$: Algorithm should be $\mathcal{O}(n^3)$
- $n \approx 50$: Algorithm should be $\mathcal{O}(n^4)$
- $n \approx 20$: Algorithm should be $\mathcal{O}(n^5)$ or $\mathcal{O}(2^n)$
- $n \approx 10$: Algorithm should be $\mathcal{O}(n^6)$ or $\mathcal{O}(n!)$

Running Time: Rule of Thumb

- Rule of Thumb: Processor can do 1M operations per second, Timelimit is 3 seconds.
- $n \approx 10^6$: Algorithm should be $\mathcal{O}(n)$
- $n \approx 10^5$: Algorithm should be $\mathcal{O}(n \log n)$
- $n \approx 10^3$: Algorithm should be $\mathcal{O}(n^2)$
- $n \approx 10^2$: Algorithm should be $\mathcal{O}(n^3)$
- $n \approx 50$: Algorithm should be $\mathcal{O}(n^4)$
- $n \approx 20$: Algorithm should be $\mathcal{O}(n^5)$ or $\mathcal{O}(2^n)$
- $n \approx 10$: Algorithm should be $\mathcal{O}(n^6)$ or $\mathcal{O}(n!)$

Running Time: Rule of Thumb

- Rule of Thumb: Processor can do 1M operations per second, Timelimit is 3 seconds.
- $n \approx 10^6$: Algorithm should be $\mathcal{O}(n)$
- $n \approx 10^5$: Algorithm should be $\mathcal{O}(n \log n)$
- $n \approx 10^3$: Algorithm should be $\mathcal{O}(n^2)$
- $n \approx 10^2$: Algorithm should be $\mathcal{O}(n^3)$
- $n \approx 50$: Algorithm should be $\mathcal{O}(n^4)$
- $n \approx 20$: Algorithm should be $\mathcal{O}(n^5)$ or $\mathcal{O}(2^n)$
- $n \approx 10$: Algorithm should be $\mathcal{O}(n^6)$ or $\mathcal{O}(n!)$

Running Time: Rule of Thumb

- Rule of Thumb: Processor can do 1M operations per second, Timelimit is 3 seconds.
- $n \approx 10^6$: Algorithm should be $\mathcal{O}(n)$
- $n \approx 10^5$: Algorithm should be $\mathcal{O}(n \log n)$
- $n \approx 10^3$: Algorithm should be $\mathcal{O}(n^2)$
- $n \approx 10^2$: Algorithm should be $\mathcal{O}(n^3)$
- $n \approx 50$: Algorithm should be $\mathcal{O}(n^4)$
- $n \approx 20$: Algorithm should be $\mathcal{O}(n^5)$ or $\mathcal{O}(2^n)$
- $n \approx 10$: Algorithm should be $\mathcal{O}(n^6)$ or $\mathcal{O}(n!)$

Storing a Graph

- Sparse graphs: adjacency list
- Dense graphs: adjacency matrix (fast lookup)

Storing a Graph

- Sparse graphs: adjacency list
- Dense graphs: adjacency matrix (fast lookup)

DFS

- Traverse graph into depth
- Similar to backtracking
- Important as subroutine for topological sorting, to find cut vertices, cut edges, or strongly connected components...

BFS

- Traverse graph into breadth
- Compute shortest distance in unweighted graph...