Stacking Boxes

- Read Exercise 1
Stacking Boxes

• Solution
  • Turns out there are only a relatively few possible configurations for the boxes
  • Determine all possible positions, then complete search on those positions
  • In particular, there can only be 1, 2, 3, or 4 stacks of boxes
Stacking Boxes

• Solution, 1-stack
  • 1000 possible positions for a singular stack
  • For each position on the number line, compute the cost of moving all boxes to the position
  • Take the minimum cost for a 1-stack solution
  • Runtime: $O(N)$
Stacking Boxes

• Solution, 2-stacks

  • Use the Sieve of Eratosthenes to compute all primes up to 1000
  
  • For each start position $i$,
    
    • Compute the cost of moving all boxes to $i$ and $i+p$ for all possible primes $p$ (such that $i+p < 1000$)
    
    • Boxes less than $(2i+p)/2$ move to $i$; otherwise to $i+p$
    
    • Precompute the sum of boxes from 0 to position $x$, use to compute the number of boxes in an interval
Stacking Boxes

- Solution, 2-stacks
  - Keep track of the minimum 2-stack solution
  - Runtime: $O(N \times N / \log(N))$
Stacking Boxes

- Solution, 3-stacks
  - For all 3-stack solutions, one pair of boxes must be distance 2 from one another

```
Stack A

Stack B

Stack C
```

\[ r = p + q \]
Stacking Boxes

• Solution, 3-stacks
  • For all 3-stack solutions, one pair of boxes must be distance 2 from one another
  • Proof by contradiction:
    • Let $p$, $q$, $r$ be primes such that $p \neq 2$, $q \neq 2$, and $r = p + q$. Then $p$ and $q$ are both odd, so $2 | (p + q)$ and $(p + q) > 4$. But then $r$ cannot be prime.
    • So either $p$ or $q$ must be 2.
Stacking Boxes

• Solution, 3-stacks
  • So, for each start position \( i \),
    • For each prime \( p \) such that \( p+2 \) is prime,
      • Compute the cost of moving boxes to stacks at \( i \), \( i+p \), \( i+p+2 \)
      • Compute the cost of moving boxes to stacks at \( i \), \( i+2 \), \( i+2+p \)
  • Keep track of the minimum 3-stack solution

• Runtime: \( O(N \times N / \log(N)) \)
  • But very few possibilities
Stacking Boxes

• Solution, 4-stacks

• There is only one possible set of distances for 4-stack solutions

\[ r = p + q \]
\[ s = q + r \]
\[ t = p + q + r \]
Stacking Boxes

• Solution, 4-stacks
  • There is only one possible set of distances for 4-stack solutions

Stack A  Stack B  Stack C  Stack D

2  q  r  s  t

r = 2 + q
s = q + r
t = 2 + q + r
Stacking Boxes

• Solution, 4-stacks

• There is only one possible set of distances for 4-stack solutions

\[ r = 2 + q \]
\[ s = q + 2 \]
\[ t = 2 + q + 2 \]
Stacking Boxes

• Solution, 4-stacks

• There is only one possible set of distances for 4-stack solutions

\[ r = 2 + 3 \]
\[ s = 3 + 2 \]
\[ t = 2 + 3 + 2 \]
Stacking Boxes

• Solution, 4-stacks
  • For each starting position $i$,
    • Compute the cost of moving boxes to $i$, $i+2$, $i+5$, and $i+7$
  • Keep track of the minimum 4-stack solution
• Runtime: $O(N)$
• Output the minimum of all solutions
Cyberline

• Read the exercise
public String lastCyberword(String cyberline) {
    String[] w = cyberline.replaceAll("-","").replaceAll("[^a-zA-Z0-9]"," "").split(" ");
    return w[w.length-1];
}
Stammering Aliens

• Read the exercise
Stammering Aliens

• Solution
  • Use a good hash function to hash the substrings and compare them
    • Too slow to store and pairwise compare strings
  • Hash function
    • hash(“babab”) = \((2 \times 31^4 + 1 \times 31^3 + 1 \times 31^2 + 1 \times 31^1 + 1 \times 3^0) \mod \text{LARGE\_PRIME}\) = 1,877,826
    • 31 is a prime number greater than 26
      • Intuitively good for reducing collisions
    • \text{LARGE\_PRIME} = 1,000,000,007
Stammering Aliens

• Solution
  
  • Modulo arithmetic properties
    
    • \((a+b) \mod p = ((a \mod p) + (b \mod p)) \mod p\)
    
    • \((a - b) \mod p = ((a \mod p) - (b \mod p)) \mod p\)
    
    • \((ab) \mod p = ((a \mod p) (b \mod p)) \mod p\)
Stammering Aliens

• Solution

  • Sweep through the substring, computing all substring hashes of length $L$
    • Precompute $31^{(L-1)} \% \text{LARGE}_{-}\text{PRIME}$ using fast exponentiation
    • Remove the leftmost character from the hash by subtracting $(\text{char}\_\text{value} \times 31^{(L-1)}) \% \text{LARGE}_{-}\text{PRIME}$
    • Add the rightmost character to the hash by multiplying by 31 and adding char_value
Stammering Aliens

- Solution
  - Maximum length of the string substring is strlen - nStrings
    - e.g., 4 cccccc
  - Try from max length down to 1
    - Too slow! :)
  - Binary search between 1 and max length
    - Turns out if you can find a solution of length L, you can find a solution of L-1 – this is a monotonic increasing function, so binary search