SIG Algorithm Challenges

Week 8: Miscellaneous Small Topics

Dane's Contact Email: dziema2@uic.edu
Github: Dane8373

Slides for this week available NOW at
https://github.com/dane8373/SIG_Algorithm_Challenges/tree/master/Week8

Join our Slack channel, #sig_algorithm_chal on the UIC ACM slack
(uicacm.slack.com)

Mock Technical Interviews Available!

Sign up here: http://bit.ly/SIG_AC_TECH

Progress So far

So far, we have covered the following topics

1) String Manipulation
2) Recursion
3) Hash Tables
4) Trees
5) Greedy Algorithms
6) Dynamic Programming
7) Graphs

These topics encompass the most common varieties of problems. Going forward, I will either cover specific interesting problems in depth, go deeper into a topic we have already covered, or introduce an advanced topic

Today, we are going to spend some time on some little tricks that don’t merit their own discussion, but are still usefu

We will just have two quick problems at the end, and will work through them together

Seach

Search problems are relatively common in interview

You have to know how to do binary search

- If you can’t write a binary search without errors, I highly recommend practicing it
- Commonly, binary search is used to find elements in a list, but it can be used more abstractly
  - E.G. searching for the size of the smallest partition of an array

Not much else to say about search, just now how to do binary search!

Bit Magic

Someitmes, Questions require bit manipulation to solve. Here are some quick pointers

Shifting and masking

- Left Shift by 1 is the same as multiply by 2. More generally, left shift by N is multiplying by 2^N
- If you want to get the 0th binary digit, you can use x&1, to get the second, you can do x&(1<<1), to get the Nth digit, you can do x&(1<<N)

xor

- Anything xor'd with something twice is unchanged. The xor operation is usually the ^ operator.
    - Example, x^y^y = x for all y.
    - Useful in some communication protocols, and also for some toy examples (E.g. given an array where all elements by one are repeated twice, find the number that isn't repeated).

Random Acts of Math

Someitmes, Questions can be solved easily with knowledge of math principles

Modular arithmetic

- A number mod 10 = the last digit of the number. E.G. 99 mod 10 = 9
  - If you divide a number by 10, then take mod 10, you get the 2nd digit. Can be iterated to extract all digits
- Converting from one base to another is generally accomplished the same way. For instance, to convert from base 10 to base 6, just take the original number mod 6 as the last digit. Then divide by 6 and add the new number mod 6 to get the next digit etc.

Summing a sequence of numbers

- The sum sequence of numbers n, n+1, n+2 … m = (m+n)*(m-n-1)/2
  - Example: 1+2+3+4….+99+100 = 101*100/2

More math

Sometimes, Questions can be solved easily with knowledge of math principles

Primes

- When checking if a number n is prime, only need to check for factors <= sqrt(n)
- Sieve of Eratosthenes is a good way to determine all primes up to some number
    - Essentially, you start with the knowledge that 2 is prime, then cross out all multiples of two as not prime, then move to the next prime number (3) and repeat the process

Greater Row Form of matrix (Often useful for hashing indexes of a matrix)

- Can convert 2D matrix indexes to 1D with the following formula: (row,col) -> #of columns * row + col.

0,0	0,1	0,2
1,0	1,1	1,2
2,0	2,1	2,2

→

0	1	2
3	4	5
6	7	8

Let’s Play a Game

Games often come up in job interviews, Games could be their own topic, however they are often too advanced for an interview

Zero Sum Games

- A Zero sum game is a game where one player doing better immediately implies that the other player did worse
    - Example: Boxing is a zero sum game, if one person scores more points in a round that explicitly means that the other person scored fewer
    - Basketball is not a zero sum game, scoring more points does not necessarily mean the opponents scored less
- Strategies for zero sum games are much easier to find than more general games, because if you greedily optimize your own score, then that strategy is more optimal

Non-Zero Sum Games

- Non zero sums games generally require a strategy like backtracking, which leads to exponential runtimes
- Generally too difficult for an interview
- If you get a game question, try to quickly prove that it is zero sum.

Let’s Play a Game

Games often come up in job interviews, Games could be their own topic, however they are often too advanced for an interview

Zero Sum Games

- A Zero sum game is a game where one player doing better immediately implies that the other player did worse
  - Example: Boxing is a zero sum game, if one person scores more points in a round that explicitly means that the other person scored fewer
  - Basketball is not a zero sum game, scoring more points does not necessarily mean the opponents scored less
- Strategies for zero sum games are much easier to find than more general games, because if you greedily optimize your own score, then that strategy is more optimal

Non-Zero Sum Games

- Non zero sums games generally require a strategy like backtracking, which leads to exponential runtimes
- Generally too difficult for an interview
- If you get a game question, try to quickly prove that it is zero sum.

Practice Problems From All Sections

We have a two dimensional matrix `A` where each value is `0` or `1`.

A move consists of choosing any row or column, and toggling each value in that row or column: changing all `0`s to `1`s, and all `1`s to `0`s.

After making any number of moves, every row of this matrix is interpreted as a binary number, and the score of the matrix is the sum of these numbers.

Return the highest possible score.

```
Input: [[0,0,1,1],[1,0,1,0],[1,1,0,0]]

Output: 39

Explanation:

Toggled to [[1,1,1,1],[1,0,0,1],[1,1,1,1]].

  0b1111 + 0b1001 + 0b1111 = 15 + 9 + 15 = 39
```

Problem Approach

Observe: No matter what, flipping the most significant bit to be a 1 is optimal

Proof: The largest possible number with 0 in the most significant bit is 0111, while that is smaller than the smallest number with 1 in the most significant bit, 1000

1: Greedily flip all the rows until they are all 1

2: Go through the columns and flip any column where there are more 0s then 1.

Practice Problems From All Sections

Write an algorithm to determine if a number is "happy".

A happy number is a number defined by the following process: Starting with any positive integer, replace the number by the sum of the squares of its digits, and repeat the process until the number equals 1 (where it will stay), or it loops endlessly in a cycle which does not include 1. Those numbers for which this process ends in 1 are happy numbers.

```
Input: 19

Output: true

Explanation:

12 + 92 = 82

82 + 22 = 68

62 + 82 = 100

12 + 02 + 02 = 1
```

Problem Approach

Pretty straight forward, just use our trick to extract digits and store the ones we have seen so far

1: Use a hashset to store all the numbers we have seen so far

2: Extract the digits and square their sum, if we have already seen this number then return false.

3: Otherwise add it to the hashset and continue

Next Week: Stump the Profs!
Meeting TBD