Datastructures And Algorithms

Interview Preparation Series

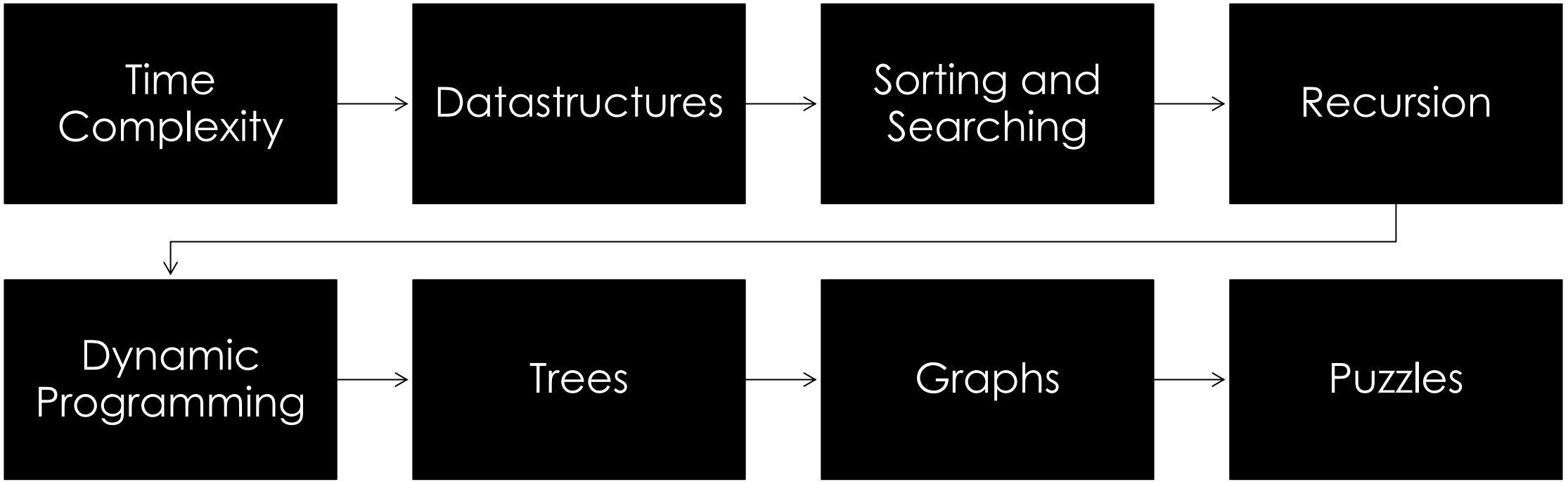
Target Audience

PROFESSIONALS LOOKING TO HONE THEIR DATASTRUCTURE SKILLS

PROFESSIONALS FROM NON-CS BACKGROUNDS WILLING TO BUILD DATASTRUCTURE SKILLS

BUDDING PROFESSIONALS WHO ARE UPSKILLING TO JOIN IT WORKFORCE

Topics


Time Complexity
Datastructures
Sorting and Searching
Recursion
Dynamic Programming
Trees
Graphs
Puzzles

Biweekly sessions

- Introduce a topic
- Discuss expectations in interviews on this topic
- Practice few problems
- Use Slack Channel for some home assignments
- Discuss on slack for any clarifications or help on the assignments and solutions

Algorithm Analysis – Time complexity

Hussaina Begum N.

$whoami

Present

- Staff Engineer @ VMware

Past

- Tech Lead @ Cisco
- Senior SDE @ Citrix
- Staff Software Engineer @ IBM ISL
- Student @ NIT Calicut

Goal of this session

- Discuss the importance of Algorithm analysis
- Learn about different time complexity notations
- Touch base on expectations in a tech interview w.r.t. DSA
- Practice some sample problems

Fundamentals

- Q: What is an algorithm ?
- Ans: Well defined computational procedure, which takes some input, produces some output – T. Corman
- Ans: Set of steps in a computer program to accomplish a task
- Real world example: Recipe for preparing a cake

Tale of two friends

Two friends A & B interviewing at a tech company

Both were asked the same problem to solve

A's program ran in 0.05s, B's program ran in 0.2s

Who should get the job?

Comparison

sum.py | rfc4718.txt | openssl_x509.c

```
1  input=[39, 28, 10, 43, 18, 55]
2
3  def sum_of_list(input):
4      sum = 0
5      for item in input:
6          sum+=item
7      return sum
8
9  print sum_of_list(input)
```

```
Darwin$time python sum.py
193

real    0m0.043s
user    0m0.017s
sys     0m0.022s
Darwin$
```

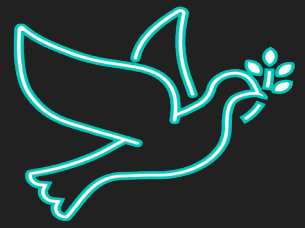
```
root@dev#time python sum.py
193

real    0m0.011s
user    0m0.006s
sys     0m0.005s
root@dev#
```

```
kali@kali:~$ time python sum.py
193

real    0m0.009s
user    0m0.006s
sys     0m0.003s
kali@kali:~$
```

Comparison



In Africa, A Pigeon Transfers Data Faster Than The Internet

JOHANNESBURG (Reuters) — A South African information technology company on Wednesday proved it was faster for them to transmit data with a carrier pigeon than to send it using Telkom, the country's leading internet service provider.

Internet speed and connectivity in Africa's largest economy are poor because of a bandwidth shortage. It is also expensive.

Local news agency SAPA reported the 11-month-old pigeon, Winston, took one hour and eight minutes to fly 50 miles from Unlimited IT's offices near Pietermaritzburg to the coastal city of Durban with a data card was strapped to his leg.

Including downloading, the transfer took two hours, six minutes and 57 seconds — the time it took for only four percent of the data to be transferred using a Telkom line.

https://www.wired.com/2009/09/in-africa-a-pigeon-transfers-data-faster-than-the-internet/

Fundamentals

- Q: What is analysis of algorithms ?
- Ans: Theoretical study of computer program's
  - Performance
  - Resource Usage
- Q: What is more important than performance?
- Ans: Correctness, usability, robustness, security, scalability, user-friendliness

Asymptotic Analysis

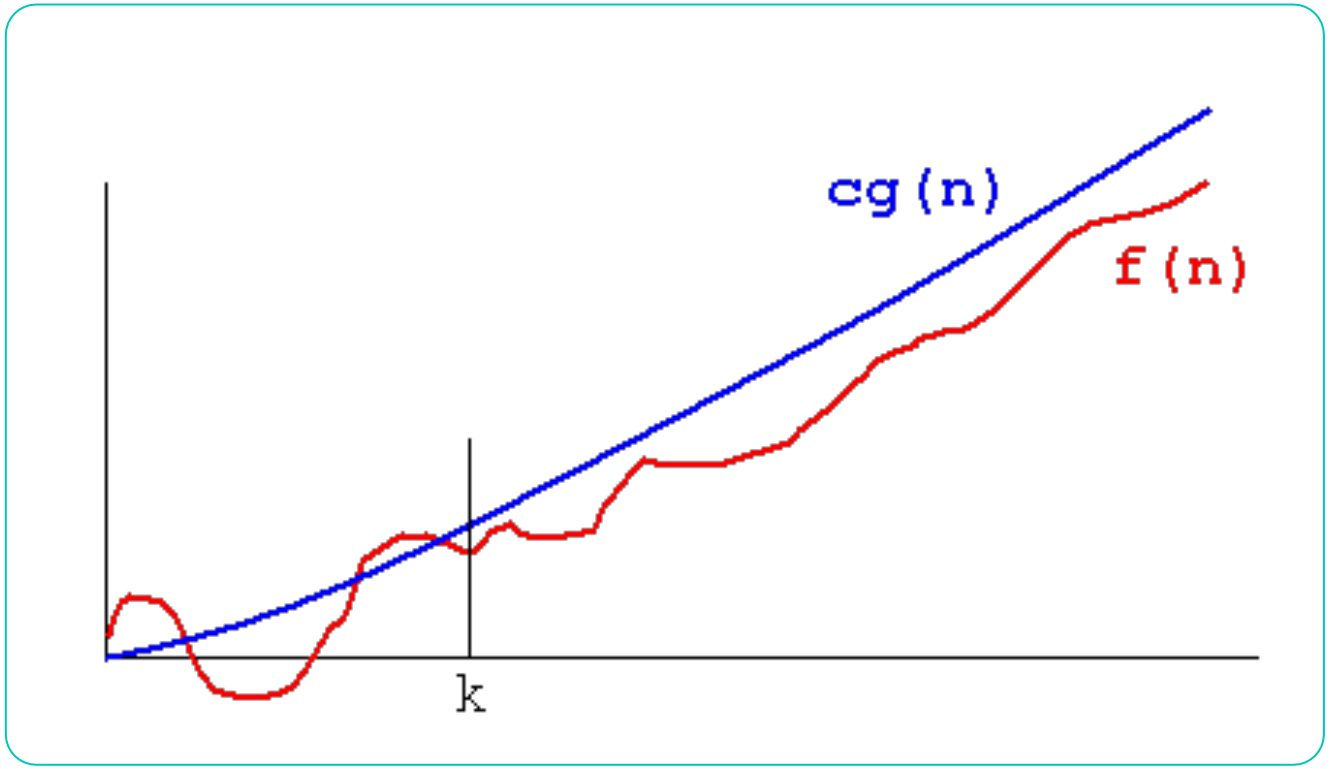
A (fuzzy) way to analyze algorithms

- Ignore machine dependent constants
- Look at rate of growth of the running time of the algorithm

Big O notation

- Upper bound
- Worst case time complexity
- f(n) = O(g(n))

  if $0 \leq f(n) \leq c.g(n)$ for all $n \geq k$, constant $c>0$, $k \geq 1$
- Drop non-dominant terms
- Drop constant coefficients

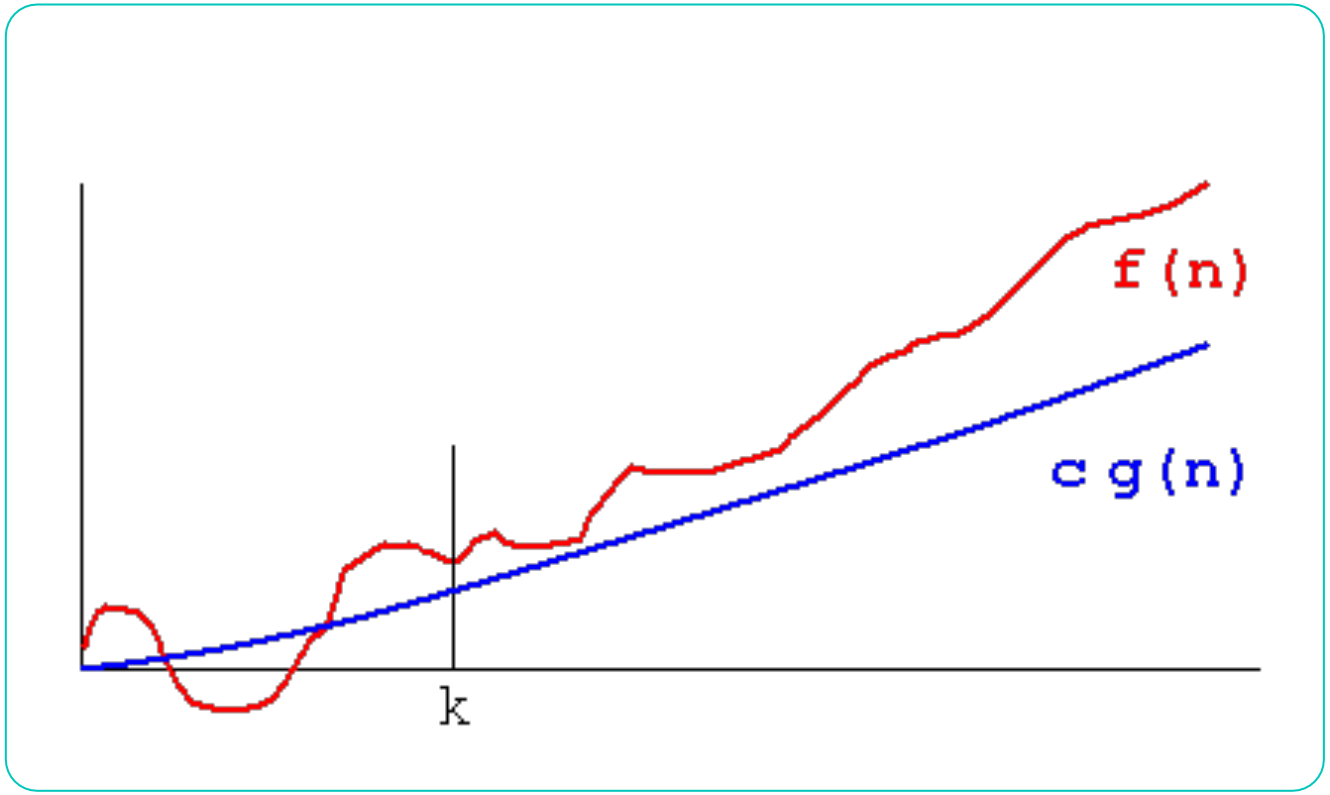


https://xlinux.nist.gov/dads/

Omega notation

- Lower bound
- Best case time complexity
- $f(n) = \Omega(g(n))$

  if $0 \leq c.g(n) \leq f(n)$ for all $n \geq k$, constant $c>0$, $k \geq 1$
- Drop non-dominant terms
- Drop constant coefficients

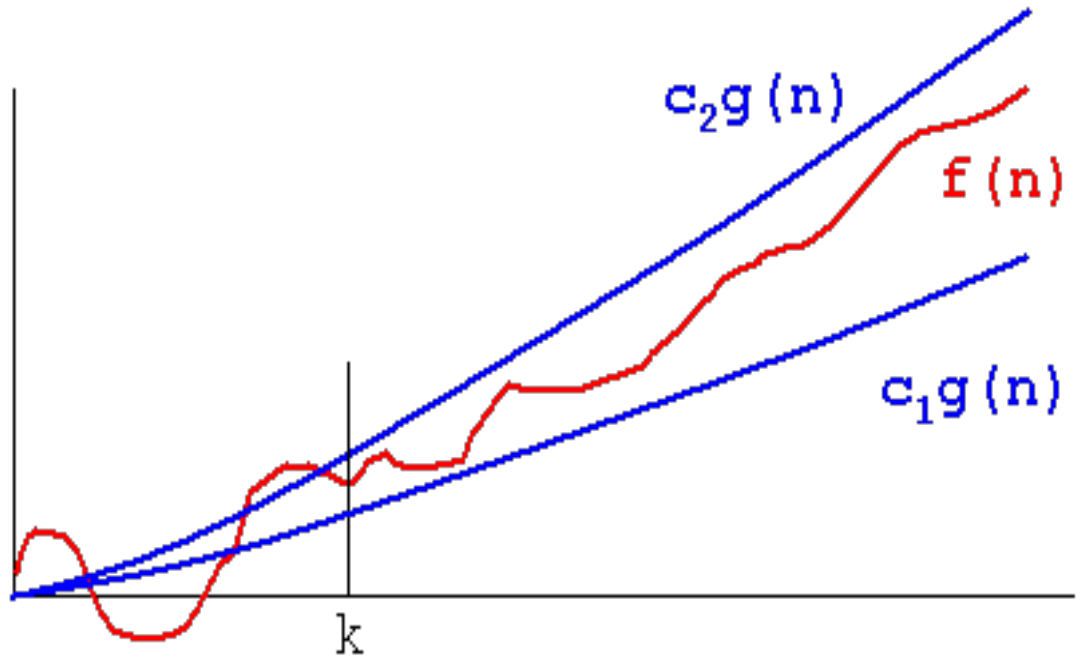


https://xlinux.nist.gov/dads/

Theta notation

- Tight bound
- Avg case time complexity
- f(n) = Θ(g(n))

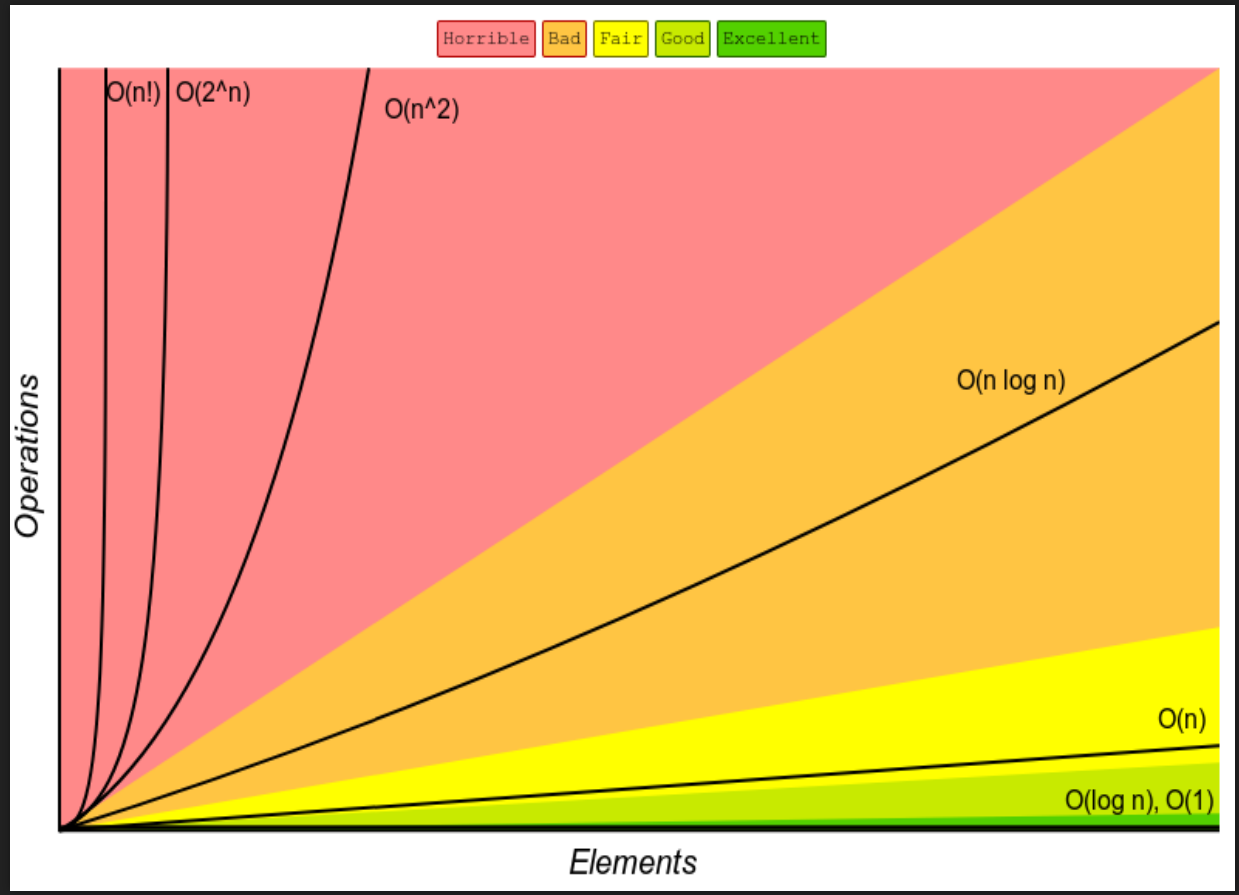
  if $0 \le c1.g(n) \le f(n) \le c2.g(n)$ for all $n \ge k$, constant c1, c2>0, $k \ge 1$
- Drop lower order terms
- Ignore leading constants



https://xlinux.nist.gov/dads/

Various Time complexities

- O(1) - Constant
- O(log n) - Logarithmic
- O(n) - Linear
- $O(n^2)$ - Quadratic
- $O(n^3)$ - Cubic
- $O(2^n)$ - Exponential
- O(n!) - Factorial



https://www.bigocheatsheet.com/

Practice

- Find the sum of all elements in the array
- Sort the elements in an array
- Find the area of a triangle
- Find an element in an sorted array
- Find the multiplication of two matrices
- Fibonacci series using recursion
- Fibonacci series using iterative
- N queens' problem

Dutch National Flag problem

Input is ['W', 'W', 'R', 'B', 'W', 'R', 'B'] and Red < White < Blue

Sort the input to ['R', 'R', ..., 'W', 'W', ..., 'B','B']

Possible Solutions -

- Sort all elements
- Count each element occurrence
- Place each element in a correct position

Space complexity

- A measure of the amount of working storage an algorithm needs to accomplish a task
- Fixed space + auxiliary space = space complexity

```
1    def add(x,y,z):
2        sum = x+y+z
3        return sum
4
5    print(add(1,2,3))
```

```
1  def nat_num(n):
2      arr = {}
3      for i in range(n):
4          arr[i] = i
5      return arr
6
7  print(nat_num(10))
```

References

- **Introduction to Algorithms (SMA 5503) -** https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-046j-introduction-to-algorithms-sma-5503-fall-2005/
- **Book - Introduction to Algorithms by CLRS**
- **Asymptotic notation -**https://www.khanacademy.org/computing/computer-science/algorithms/asymptotic-notation/a/asymptotic-notation

Resources

- https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-006-introduction-to-algorithms-fall-2011/
- https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-046j-introduction-to-algorithms-sma-5503-fall-2005/
- **Book - Introduction to Algorithms by CLRS**

- Thank you, see you in the next session!