COL351: Analysis and Design of Algorithms

Ragesh Jaiswal, CSE, IITD

Recap. of Data Structures and Algorithms

Recap.

- What is an algorithm?

Recap.

- What is an algorithm?
  - A step-by-step way of solving a problem.
- How do we measure the performance of an algorithm?

Recap.

- What is an algorithm?
  - A step-by-step way of solving a problem.
- How do we measure the performance of an algorithm?
- Main ideas for performance measurement:
  - Worst-case analysis: Largest possible running time over all input instances of a given size n and then see how this function scales with n.
  - Asymptotic order of growth: The worst-case running time for large n (e.g., $T(n) = 5n^3 + 3n^2 + 2n + 10$)

Recap.

- What is an algorithm?
  - A step-by-step way of solving a problem.
- How do we measure the performance of an algorithm?
- Main ideas for performance measurement:
  - Worst-case analysis: Largest possible running time over all input instances of a given size n and then see how this function scales with n.
  - Asymptotic order of growth: The worst-case running time for large n (e.g., $T(n) = 5n^3 + 3n^2 + 2n + 10$)

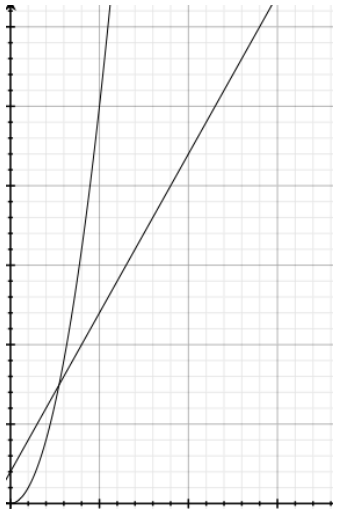

Figure : Plot of n^2 and $2n + 2$

Recap.

- What is an algorithm?
  - A step-by-step way of solving a problem.
- How do we measure the performance of an algorithm?
- Main ideas for performance measurement:
  - Worst-case analysis: Largest possible running time over all input instances of a given size n and then see how this function scales with n.
  - Asymptotic order of growth: The worst-case running time for large n (e.g., $T(n) = 5n^3 + 3n^2 + 2n + 10$)
- Asymptotic order of growth (O, Ω, Θ):
  - $T(n)$ is $O(f(n))$ (or $T(n) = O(f(n))$) iff there exists constants $c > 0, n_0 \geq 0$ such that for all $n \geq n_0$, we have $T(n) \leq c \cdot f(n)$.

Recap.

- Growth rates:
  - Arrange the following functions in ascending order of growth rate:
    - n
    - $2^{\sqrt{\log n}}$
    - $n^{\log n}$
    - $2^{\log n}$
    - $n/\log n$
    - n^n

Recap.

Problem

Given a positive integer, check if the number is prime.

Solution

isPrime(n)

- For $i = 2$ to $(n - 1)$
  - If $(n\%i == 0)$ then output "no"
- output "yes"

- What is the running time of this algorithm?
- Is this an efficient algorithm?

Recap.

Problem

Given a sorted array, check if it contains a given number n.

- Solution: Binary search
  - What is the running time?
- Alternate solution: Divide the array into **3** parts and recursively find the given element in one of the three parts.
  - What is the running time?

Recap.

Problem

Given a sorted array, check if it contains a given number n.

- Solution: Binary search
  - What is the running time?
- Alternate solution: Divide the array into **3** parts and recursively find the given element in one of the three parts.
  - What is the running time?
- Which one is the better algorithm?

Recap.

- Problem: Given two n-bit numbers, multiply them.
- What is the running time of the naïve algorithm?
- Is it possible to get a better algorithm?

End