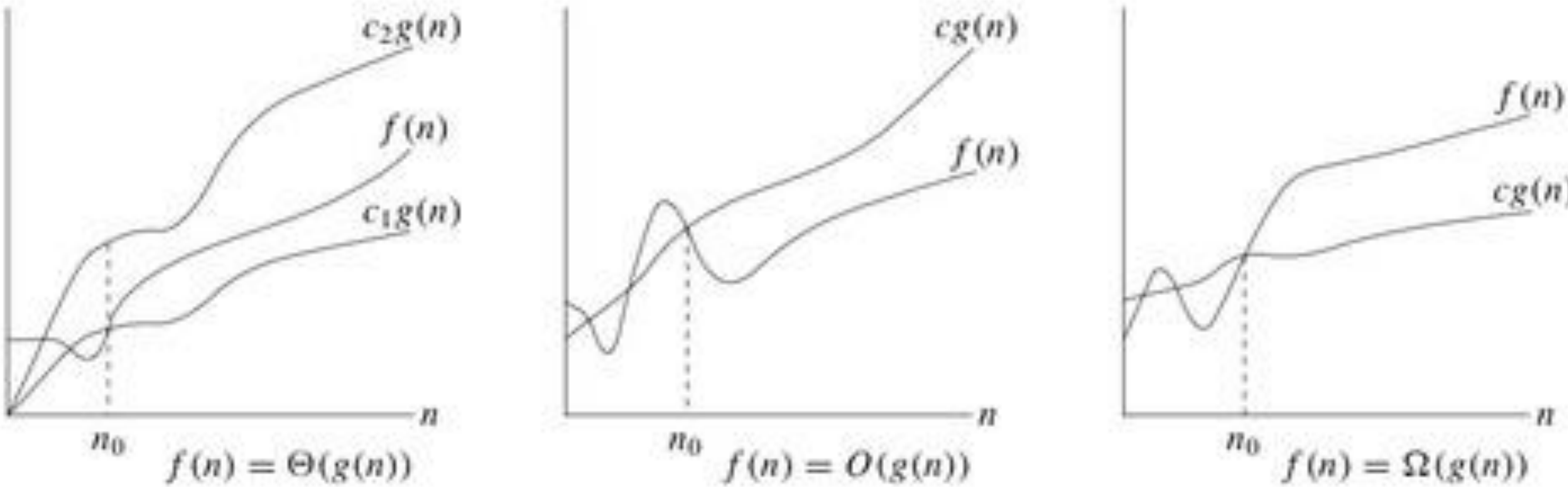
3 Notation Summary



Big O notation

$$O(f(n)) : \exists\, c_0,\ n_0 \text{ s.t. } |a(n)| \le c_0 f(n), \forall n \ge n_0.$$

θ notation

$$\Theta(f(n)) : \exists\, c_0,\ c_1,\ n_0 \text{ s.t. } c_0 f(n) \le |a(n)| \le c_1 f(n), \forall n \ge n_0.$$
$$\Theta(f(n)) = O(f(n)) \cap \Omega(f(n)).$$

Big Omega notation

$$\Omega(f(n)) : \exists\, c_0,\ n_0 \text{ s.t. } |a(n)| \ge c_0 f(n), \forall n \ge n_0.$$

- Proof by definition
- Proof by induction
- Proof by contradiction

Mathematical Induction

Principle of Mathematical Induction: To prove that $P(n)$ is true for all positive integers n, we complete these steps:

BASIS STEP: Show that $P(n_0)$ is true.

INDUCTIVE STEP: Show that $P(k) \to P(k+1)$ is true for all positive integers $k \geq n_0$.

To complete the inductive step, assuming the inductive hypothesis that $P(k)$ holds for an arbitrary integer $k \geq n_0$, show that must $P(k+1)$ be true.

Examples:

1-n is in O(2^n)

- Prove that n<=$c.\,2^n$
- By definition : assign some correct value in c and n_0 (2,1)
- By induction:
  - n=1, 1<=$c.\,2^1$
  - n=k, k<=$c.\,2^k$
  - n=k+1, k+1<=$c.\,2^{k+1}$

- 2- Sort the following functions from slowest to fastest in terms of their growth. (Hint: Try to put all the functions in the same form, for example exponential form)

  $\log(n!)$, $n^{\log n}$, $n^{1.54}$, $2^{\log n/\log\log n}$, $(\log n)^{(\log n)}$

- Solution:

The correct ordering, from smallest to largest, is:

$$2^{\log n/\log\log n}, \log(n!), n^{1.54}, (\log n)^{(\log n)}, n^{\log n}$$

To prove the correctness of this ordering, we may proceed by putting all the functions in exponential form:

- $n^{\log n} = 2^{\log n \log n} = 2^{\log^2 n}$
- $n^{1.54} = 2^{\log n^{1.54}} = 2^{1.54 \log n}$
- $2^{\log n/\log\log n}$ is already in the proper form
- $(\log n)^{(\log n)} = 2^{\log(\log n)^{(\log n)}} = 2^{\log n \cdot \log\log n}$

$\log(n!)$ takes a bit more work. We can easily find lower and upper bounds for it:
$\log(n!) \geq \log(2^n) = n = 2^{\log n}$
$\log(n!) \leq \log(n^n) = n \log n = 2^{\log n + \log\log n}$
We can now easily see the correct ordering of these functions as below (the limit of the ratio between any function that appears later to a function that appears earlier in the ordering, goes to infinity):

$$2^{\log n/\log\log n}, \underbrace{2^{\log n}, 2^{\log n + \log\log n}}_{\log(n!)}, 2^{1.54 \log n}, 2^{\log n \cdot \log\log n}, 2^{\log^2 n}$$

- 3- calculate the running time funtion below

```
for (i=2*n; i>=1; i=i-1)
        for (j=1; j<=i; j=j+1)
                for (k=1; k<=j; k=k*3)
                        print("hello")
```

- Solution:
- $f(x) = \sum_{i=1}^{2n} \left(\sum_{j=1}^{i} \frac{j}{3} + 1 \right)$
- $\frac{j*(j+1)}{6}$+j
- $\frac{1}{6} \sum_{i=1}^{2n} (j^2 + 7j\ )$

- 4

1- For each f(n) and g(n) below, it is either f(n) is in O(g(n)) or g(n) is in O(f(n)). Using only the definition of O notation and the proof methods you learned in discrete math class, prove which one is correct.

- $f(n) = (n^2 - n)/2$, $g(n) = 4n$
- $f(n) = n+\log n$, $g(n) = n$*SquareRoot(n)
- $f(n) = 2(\log n)^2$, $g(n) = \log n + 1$

- Solution a

f(n)=$\frac{(n^2-n)}{2}$, g(n)=4n

For c=1, n_0=9

4n<=$\frac{(n^2-n)}{2}$

…

9<=n

g(n)=O(f(n))

- Solution b

f(n)=$n + logn$, g(n)= $n\sqrt{n}$

$$lim_{n=\infty} \frac{n\sqrt{n}}{n + logn}$$

$$lim_{n=\infty} \frac{3/2n^{1/2}}{1+1/n} =\infty$$

g(n)=O(f(n))

C

f(n)=2*$logn^2$, g(n)= $logn + 1$

$$lim_{n=\infty} \frac{2{*}logn^2}{logn + 1}$$

$$lim_{n=\infty} \frac{4{*}logn{*}1/n}{1/n} = \infty$$

g(n)=O(f(n))

4

Bilgisayar Mühendisliği okuyan üç arkadaş, karşılıklı olarak birbirlerinin bilgisayarları çökertmek için iddiaya girerler ve birbirlerinin bilgisayarlarına birer program yazarlar.

Birinci arkadaşın yazdığı program f(n)= n^2 zaman karmaşıklığına sahiptir. İkinci arkadaşın yazdığı program ise g(n)=2^n zaman karmaşıklığına sahiptir.Son olarak üçüncü arkadaşın yazdığı program h(n)=n! zaman karmaşıklığına sahiptir. Yeterince büyük verilen n sayısı için kimin yazdığı programın daha önce diğer bilgisayarları çökerteceğini bulunuz. Ayrıca verilen çalışma zamanlarını gerekli asimptotik

gösterimleri kullanarak karşılaştırınız.

- f(n)= n^2
- g(n)=2^n
- h(n)=n!

5

Aşağıda verilen algoritma bir A bilgisayarında girilen bir n sayısı için 100 saniye sürmektedir.

a) Aynı bilgisayarda 5n sayısı girdi olarak verildiğinde algoritma kaç saniye sürer?

```
int F(int M[a][b])
int min = M[0][0]
for(int i = 0 ; i < a ; i++)
      for(int j = 0 ; j < b ; j++)
            if (min > M)
                  min = M[i][j]
```

Proof by contradiction

- To clarify the process of proof by contradiction further, let's break it down into steps. When using proof by contradiction, we follow these steps.
- Assume your statement to be false.
- Proceed as you would with a direct proof.
- Come across a contradiction.
- State that because of the contradiction, it can't be the case that the statement is false, so it must be true.

6

- $\sqrt{n}$ is *not* in $\Omega(n)$