Discrete Structures

Dr. Syed Faisal Bukhari
Associate Professor
Department of Data Science (DDS), Faculty of Computing and Information Technology (FCIT), University of the Punjab (PU)

Text book

Discrete Mathematics and Its Application, 7^{th} Edition

Kenneth H. Rosen

References

Chapter 3

1. Discrete Mathematics and Its Application, 7^h Edition
by
Kenneth H. Rose

2. Discrete Mathematics with Applications
by
Thomas Koshy

3. Discrete Mathematical Structures, CS 173
by
Cinda Heeren, Siebel Center

4. https://www.cs.cornell.edu/courses/JavaAndDS/files/constantTime.pdf

5. Discrete Mathematics for Computer Science by Gary Haggard

These slides contain material from the above resources.

Constant time

Constant time: An operation or method takes constant time if the time it takes to carry it out does not depend on the **size of its operands**.

For example, an **array element reference b[i]** takes **constant time**, but **printing out all elements of array b** is **not constant time** but instead takes time proportional to the **size of b.**

Also, this assignment takes **constant time**:

b[i]= b[i] + 2

Example Give **big-O estimates** for the **factorial function** and the **logarithm of the factorial function**, where the factorial function **$f(n) = n!$** is defined by

$n! = 1 \times 2 \times 3 \ldots n$, whenever n is a positive integer, and

$0! = 1$.

For example,

$1! = 1$, $2! = 1 \times 2 = 2$, $3! = 1 \times 2 \times 3 = 6$, $4! = 1 \times 2 \times 3 \times 4 = 24$.

Note that the function **n! grows rapidly**.

For instance, **20! = 2,432,902,008,176,640,000**

Solution:

We have to show $f(n) = n!$ is $O(n^n)$

$|f(n)| \leq C|g(n)|$ whenever $n > k$.

$n! = 1 \times 2 \times \ldots n$

$n! \leq n \times n \times n \ldots n$, where $k > \mathbf{1}$

$0 \leq n! \leq n \times n \times n \ldots n = n^n$

$\Rightarrow n! \leq n^n$

$\Rightarrow n! \leq \mathbf{1} \times n^n$

$= Cg(n)$

n	$n! \leq n^n$
1	$1! \leq 1^2$ (true)
2	$2! \leq 2^2$ (True)
3	$3! \leq 3^2$ (True)
⋮	⋮

Here **$f(n) = n!$** and **$g(n) = n^n$**

Consequently, we can take C = 1and k = 1 as witnesses to show that $f(n)$ is $O(n^n)$

$\because n! \leq n^n$

Taking log on both sides

$\Rightarrow \log n! \leq \log n^n$

$\Rightarrow \log n! \leq n \log n$ $\because \boldsymbol{\log m^n = n \times \log m}$

This implies that **log n!** is **O(n log n),** again taking C = 1 and k = 1 as witnesses.

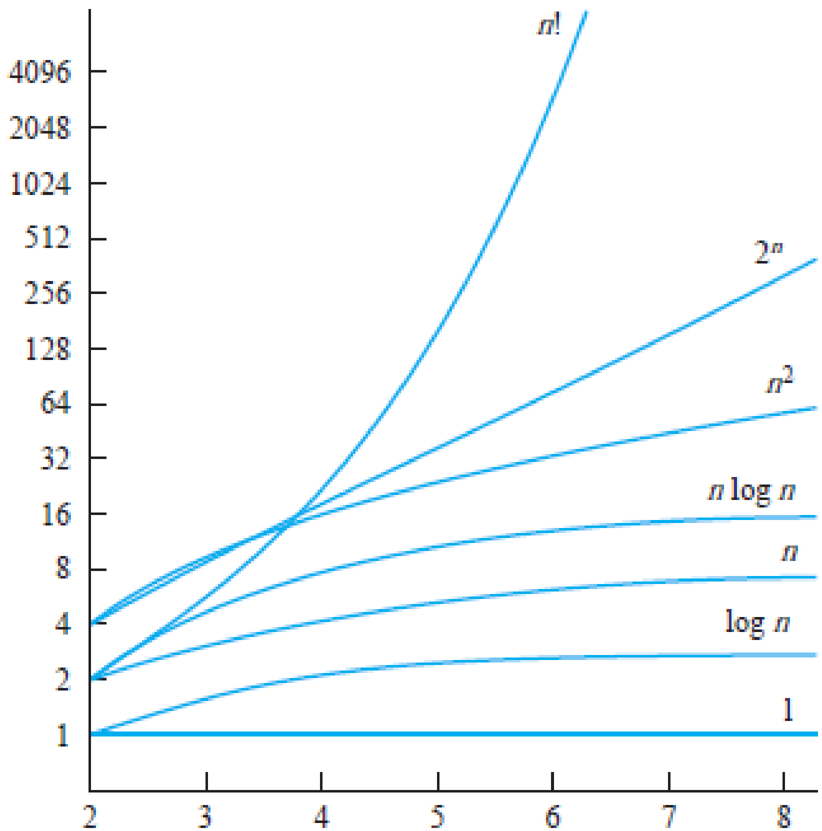
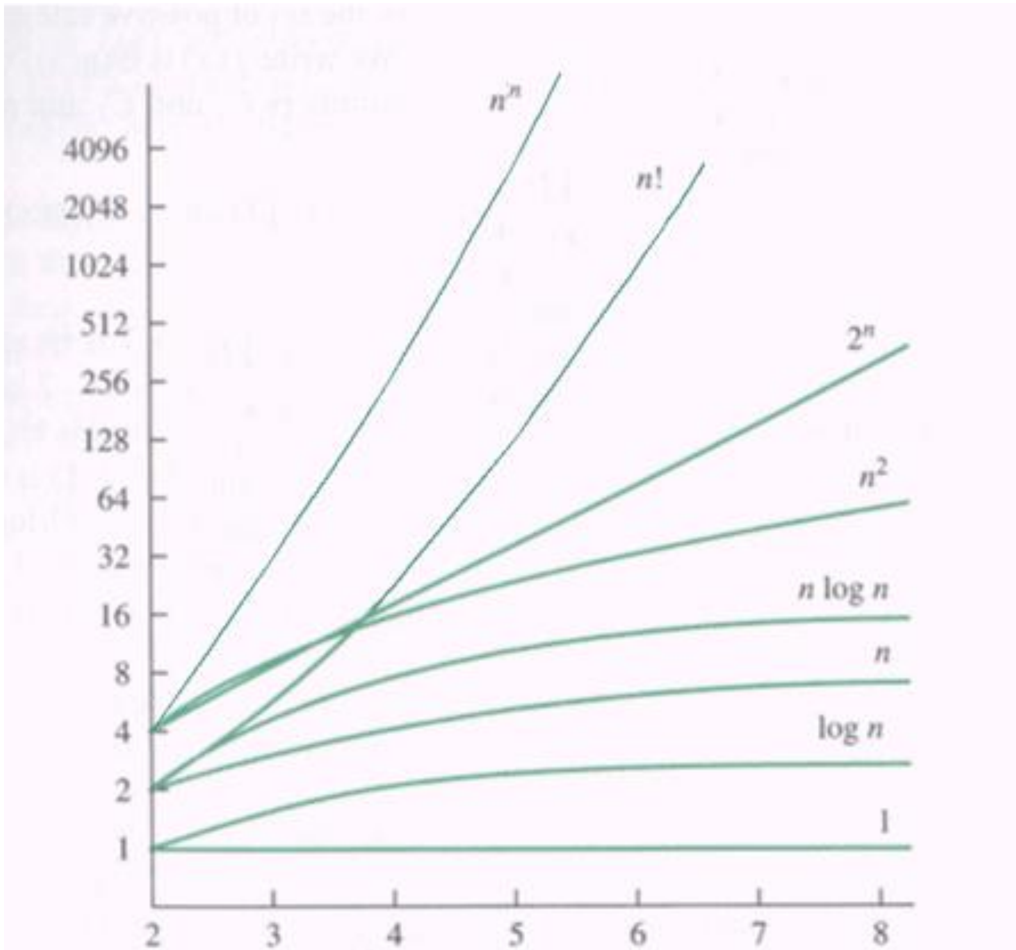


FIGURE 3 **A Display of the Growth of Functions Commonly Used in Big-O Estimates.**



n^n
$n!$
2^n
n^2
$n \log n$
n
$\log n$
1
4096
2048
1024
512
256
128
64
32
16
8
4
2
1
2
3
4
5
6
7
8

Complexity Comparisons for Various Functions

- To see the difference in the **time requirement** for processing **data sets of arbitrary size**, we will assume **a single machine cycle** will require **10^{-6} seconds** to be completed.

- **Table** on the next slide gives the **time required to process a data set of size n**, for six different values of n, if it takes $\log_2(n)$ *(n, n^2, n^5 ,* and 2^n, respectively) machine cycles to make the computation.

- For example, in the column labeled n^2 for the row labeled n = 100, 100^2 operations are needed to complete execution. The time is **$(10^2)^2 10^{-6}$ seconds = 10^{-2} seconds.**

Complexity Comparisons for Various Functions

$F(n)$	$log_2(n)$	n	n^2	n^5	2^n
$n = 10$	3×10^{-6} sec	10^{-5} sec	10^{-4} sec	0.1 sec	10^{-3} sec
$n = 20$	4×10^{-6} sec	2×10^{-5} sec	4×10^{-4} sec	3 sec	1 sec
$n = 50$	6×10^{-6} sec	5×10^{-5} sec	3×10^{-3} sec	5 min	36 yrs
$n = 100$	7×10^{-6} sec	10^{-4} sec	10^{-2} sec	3 hrs	4×10^{16} yrs
$n = 1000$	1×10^{-5} sec	10^{-3} sec	1 sec	32 yrs	3.9×10^{287} yrs
$n = 100,000$	2×10^{-5} sec	0.1 sec	2.7 hrs	3×10^{11} yrs	$> 10^{30,089}$ yrs

What Does Machine Cycle Mean?

- A machine cycle consists of the steps that a computer’s **processor executes whenever it receives a machine language** instruction.

- It is the most basic CPU operation, and modern CPUs are able to perform millions of machine cycles per second.

- The cycle consists of three standard steps: fetch, decode and execute. In some cases, store is also incorporated into the cycle.

Show that $7x^2$ is $O(x^3)$.

Solution:

$7x^2$ is $O(x^3)$.

$|f(x)| \leq C|g(x)|$ whenever $x > k$.

$f(x) = 7x^2$

$g(x) = x^3$

We observe that we can readily estimate the size of f (x) when **$x > 7$**

$\because 7x^2 < x^3$. when $x > 7$.

$0 \leq 7x^2 \leq x^3$

$\Rightarrow 7x^2 \leq \mathbf{1} \times x^3$

$\Rightarrow f(x) \leq Cg(x)$

Consequently, we can take ***C* = 1** and ***k* = 7** as witnesses to establish

Show that $7x^2$ is $O(x^3)$.

Alternative solution:

$7x^2$ is $O(x^3)$.

$|f(x)| \le C|g(x)|$ whenever $x > k$.

$f(x) = 7x^2$

$g(x) = x^3$

We observe that we can readily estimate the size of f (x) when x > 1

$\because 7x^2 < 7x^3$. **when x > 1**.

$0 \le 7x^2 \le 7x^3$

$\Rightarrow 7x^2 \le \mathbf{7 \times x^3}$

$\Rightarrow f(x) \le Cg(x)$

Consequently, we can take ***C* = 7** and ***k* = 1** as witnesses to establish

Show that n^2 is $O(n)$.

Solution:

n^2 is $O(n)$.

$|f(n)| \leq C|g(n)|$ whenever $n > k$.

$f(n) = n^2$

$g(n) = n$

We have to show that

$n^2 \leq Cn$

Dividing both sides by n

$\Rightarrow n \leq C$

We cannot find any *C* and *k* as witnesses

Note: C and k are constants, whereas k is a positive real number and C is a real number

Example: Find Big-oh notation of $1 + 2 + 3 + \ldots + n$

Let $f(n) = 1 + 2 + 3 + \ldots + n$

We have to show that $f(n) = 1 + 2 + 3 + \ldots + n$ is $\mathbf{O(n^2)}$

$\because |f(n)| \leq C|g(n)|$ **whenever** $n > k$.

$1 < n^2, \; 2 < n^2, \; 3 < n^2, \ldots$ so on when $n > 1$

$$
\begin{aligned}
f(n) &= 1 + 2 + 3 + \ldots + n \\
&\leq n + n + n + \ldots + n \\
&\leq n \times n \\
&\leq 1 \times n^2
\end{aligned}
$$

$\Rightarrow f(n) \leq Cg(n)$

Consequently, we can take $C = 1$ and $k = 1$ as witnesses to establish

Show that logn is O(n)

We will prove by mathematical induction

$n < 2^n$

$\Rightarrow f(x) \leq Cg(x)$

Consequently, we can take $C = 1$ and $k = 1$ as witnesses to establish

We conclude that **n** is **$O(2^n)$**

$n < 2^n$

Taking log on both sides with base 2

$\lg n < \lg 2^n$

$\lg n < n$**$\lg 2$** $\qquad$ **$\because \lg 2 = 1$**

$\lg n < n$

$\Rightarrow f(x) \leq Cg(x)$

Consequently, we can take $C = 1$ and $k = 1$ as witnesses to establish. We conclude that **lgn is O(n)**

Cont.

If we have logarithms to a base *b*, where *b* is different from 2, we still have **$\log_b n$** is ***O(n)***

Because

$\because \lg n < n$

$$\log_b n = \frac{\log n}{\log b} < \frac{n}{\log b} \qquad \because \log_b n = \frac{\log_c n}{\log_c b}$$

$$\log_b n = \frac{\log n}{\log b} < \frac{1}{\log b} \times n$$

whenever *n* is a positive integer. We take **$C = 1/\log b$** and $k = 1$ as witnesses.

- $n! = O(n^n)$
- $\log(n!) = O(n \log n)$.
- $\log_b n = O(n)$
- $n = O(2^n)$
- $f(n) = 1 + 2 + 3 + \ldots + n$ is $O(n^2)$

The Growth of Combinations of Functions

Theorem Suppose that $f_1(x)$ is $O(g_1(x))$ and that $f_2(x)$ is $O(g_2(x))$. Then $(f_1 + f_2)(x)$ is $O(\max(|g_1(x)|, |g_2(x)|))$.

Corollary Suppose that $f_1(x)$ and $f_2(x)$ are both $O(g(x))$. Then $(f_1 + f_2)(x)$ is $O(g(x))$.

Theorem Suppose that $f_1(x)$ is $O(g_1(x))$ and $f_2(x)$ is $O(g_2(x))$. Then $(f_1f_2)(x)$ is $O(g_1(x)g_2(x))$.

Example Give a **big-O estimate** for $f(n) = 3n \log(n!) + (n^2 + 3) \log n$, where n is a positive integer.

Solution: for $f(n) = 3n \log(n!) + (n^2 + 3) \log n$

First, the product $3n \log(n!)$ will be estimated.

$\log(n!) = O(n \log n)$.

$3n$ is $O(n)$

Suppose that $f_1(x)$ is $O(g_1(x))$ and $f_2(x)$ is $O(g_2(x))$. Then $(f_1f_2)(x)$ is $O(g_1(x)g_2(x))$.

$\Rightarrow 3n \log(n!) = O(n^2 \log n)$.

$(n^2 + 3) \log n$ will be estimated:

Because $(n^2 + 3) < 2n^2$ when $n > 2$, it follows that

$n^2 + 3 = O(n^2)$.

$\Rightarrow (n^2 + 3) \log n = O(n^2 \log n)$.

Cont.

Theorem Suppose that $f_1(x)$ is $O(g_1(x))$ and that $f_2(x)$ is $O(g_2(x))$. Then $(f_1 + f_2)(x)$ is $O(\max(|g_1(x)|, |g_2(x)|))$.

$\therefore f(n) = 3n \log(n!) + (n^2 + 3) \log n$

is $O(\max(n^2 \log n, n^2 \log n))$ is $O(n^2 \log n)$.

Example: Let $f(n) = 6n^2 + 5n + 7\lg n!$ Estimate the growth of $f(n)$

Solution:

$f(n) = 6n^2 + 5n + 7\lg n!$

$6 = O(1)$

$6n^2 = O(n^2)$

$5 = O(1)$

$5n = O(n)$

$\because$ **$(f_1 + f_2)(x)$** is **$O(\max(|g_1(x)|, |g_2(x)|))$**

$6n^2 + 5n = O(\max(n^2, n)) = O(n^2)$

$7 = O(1)$

$\lg n! = O(n\lg n)$

$\because$ **$(f_1 f_2)(x)$ is $O(g_1(x)g_2(x))$**

$7\lg n! = O(1.n\lg n) = O(n\lg n)$

$\because$ **$(f_1 + f_2)(x)$** is **$O(\max(|g_1(x)|, |g_2(x)|))$**

$6n^2 + 5n + 7\lg n!$ is $O($**max**$(n^2, n\lg n)) = O(n^2)$

big-Omega (Ω)

Let **f** and **g** be functions from the set of integers or the set of real numbers to the set of real numbers. We say that f(x) is **Ω(g(x))** if there are positive constants C and k such that

$|f(x)| \geq C|g(x)|$ whenever $x > k$. [This is read as "f (x) is big-Omega of g(x)."]

Note: There is a strong connection between big-O and big-Omega notation. In particular, **f(x) is Ω (g(x))** if and only **if g(x) is O(f(x))**

Example The function **$f(x) = 8x^3 + 5x^2 + 7$** is $\Omega(g(x))$, where $g(x)$ is the function $g(x) = x^3$.

$|f(x)| \geq C|g(x)|$ whenever $x > k$.

This is easy to see because **$f(x) = 8x^3 + 5x^2 + 7 \geq 8x^3$** for all positive real numbers x.

This is equivalent to saying that $g(x) = x^3$ is $O(8x^3 + 5x^2 + 7)$, which can be established directly by turning the inequality around.

big-Theta(Θ)

- Let f and g be functions from the set of integers or the set of real numbers to the set of real numbers. We say that f(x) is ***Θ*(g(x)) if f(x) is O(g(x)) and f(x) is Ω(g(x)).** When f(x) is Θ(g(x)) we say that "f is big-Theta of g(x)'', that f(x) is of order g(x), and that f(x) and g(x) are of the same order.

- When f(x) is Θ(g(x)), it is also the case that g(x) is Θ(f (x)). Also note that f (x) is Θ(g(x)) if and only if f (x) is O(g(x)) and g(x) is O(f (x)).

- **f (x) is Θ(g(x))** if and only if there are real numbers C_1 and C_2 and a positive real number k such that $\mathbf{C_1|g(x)| \leq |f(x)| \leq C_2|g(x)|}$ whenever $x > k$. The existence of the constants C_1, C_2, and k tells us that f (x) is Ω(g(x)) and that f (x) is O(g(x)), respectively

Example Show that $3x^2 + 8x \log x$ is $\Theta(x^2)$.

Solution:

f(x) = $3x^2 + 8x \log x$

$C_1|g(x)| \leq |f(x)| \leq C_2|g(x)|$ whenever $x > k$

For O:

$|f(x)| \leq C_2|g(x)|$ whenever $x > k$.

$x \log x < x^2$ whenever $x > 1$

$0 \leq 3x^2 + 8x \log x \leq 3x^2 + 8x^2 \leq 11x^2$ whenever $x > 1$

$3x^2 + 8x \log x \leq 11x^2$

$3x^2 + 8x \log x \leq Cg(x)$ whenever $x > 1$

$C_2 = 11, k = 1$

Note: C_2 and k are constants.

We have to show that $3x^2 + 8x \log x$ is $\Theta(x^2)$.

f(x) = $3x^2 + 8x \log x$

For Ω:

$|f(x)| \geq C_1|g(x)|$ whenever $x > k$.

$3x^2 + 8x \log x \geq$ **3**x^2 whenever $x > 1$.

$3x^2 + 8x \log x \geq C_2|g(x)|$

$C_1 = 3$, $k = 1$

Note: C_1 and k are constants.

$3x^2 + 8x \log x$ is $\Theta(x^2)$.

Theorem Let $\mathbf{f(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0}$, where $a_0, a_1, \ldots, a_n$ are real numbers with $a_n \neq 0$. Then ***f(x)*** is of order $\mathbf{x^n}$.

Example The polynomials

$3x^8 + 10x^7 + 221x^2 + 1444$,

$x^{19} - 18x^4 - 10{,}112$,

and

$-x^{99} + 40{,}001x^{98} + 100{,}003x$

are of orders x^8, x^{19}, and x^{99}, respectively.

Suggested Readings

Chapter 3

3.1 Algorithms

3.2 The Growth of Functions