THE GROWTH OF FUNCTIONS

Rosen 6th Ed., 3.2, 3.3

Logarithms Review

\[ \log_b x = y \iff b^y = x \]
\[ \log_{10} 100 = 2 \iff 10^2 = 100 \]
\[ \log_2 16 = 4 \iff 2^4 = 16 \]

Basic Rules of Logarithms

\[ \log_z (xy) = \log_z (x) + \log_z (y) \]
\[ \log_z (x/y) = \log_z (x) - \log_z (y) \]
\[ \log_z (x^y) = y \log_z (x) \]
If \( x = y \) then \( \log_z (x) = \log_z (y) \)

Growth

- If \( f \) is a function from \( \mathbb{Z} \) or \( \mathbb{R} \) to \( \mathbb{R} \), how can we quantify the rate of growth and compare rates of growth of different functions?
- Possible problem: Whether \( f(n) \) or \( g(n) \) is larger at any point may depend on value of \( n \).
For example: \( 100n > n^2 \) if \( n < 100 \)

How to quantify growth as \( n \) gets larger?
Big-O Notation

- 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 \( O(g(x)) \) if there are constants \( C \in \mathbb{N} \) and \( k \in \mathbb{R} \) such that \( |f(x)| \leq C |g(x)| \) whenever \( x > k \).
- We say “\( f(x) \) is big-oh of \( g(x) \)”.
- The intuitive meaning is that as \( x \) gets large, the values of \( f(x) \) are no larger than a constant times the values of \( g(x) \), or \( f(x) \) is growing no faster than \( g(x) \).

Show that \( 3x^3+2x^2+7x+9 \) is \( O(x^3) \)

Proof: We must show that \( \exists \) constants \( C \in \mathbb{N} \) and \( k \in \mathbb{R} \) such that \( |3x^3+2x^2+7x+9| \leq C |x^3| \) whenever \( x > k \).

Choose \( k = 1 \) then
\[
3x^3+2x^2+7x+9 \leq 3x^3+2x^3+7x^3+9x^3 = 21x^3
\]

So let \( C = 21 \).

Then \( 3x^3+2x^2+7x+9 \leq 21x^3 \) when \( x \geq 1 \).

General Rules

- Multiplication by a constant does not change the rate of growth. If \( f(n) = kg(n) \) where \( k \) is a constant, then \( f \) is \( O(g) \) and \( g \) is \( O(f) \).
- Addition of smaller terms does not change the rate of growth. If \( f(n) = g(n) + \text{smaller terms} \), then \( f \) is \( O(g) \) and \( g \) is \( O(f) \).

Ex.: \( f(n) = 4n^6 + 3n^5 + 100n^2 + 2 \) is \( O(n^6) \).

General Rules (cont.)

- If \( f_1(x) \) is \( O(g_1(x)) \) and \( f_2(x) \) is \( O(g_2(x)) \), then \( f_1(x)f_2(x) \) is \( O(g_1(x)g_2(x)) \).
- Examples:
  - \( 10x \log_2 x \) is \( O(x \log_x 2) \)
  - \( n!6n^3 \) is \( O(n!n^3) \)
Examples

• $f(x) = 10$ is $O(1)$
• $f(x) = x^2 + x + 1$ is $O(x^2)$
• $f(x) = 2x^5 + 100x^3 + x \log x$ is $O(x^5)$
• $f(x) = 2^n + n^{10}$ is $O(2^n)$

How would you prove that $2^n$ is bigger than $n^{10}$?

Show that $n!$ is $O(n^n)$

Proof: We must show that there exist constants $C \in \mathbb{N}$ and $k \in \mathbb{R}$ such that $n! \leq C \cdot n^n$ whenever $n > k$.

$n! = n(n-1)(n-2)\ldots(3)(2)(1) \leq n(n)(n)\ldots(n)(n)(n)$ \hspace{1cm} $n$ times

$= n^n$

So choose $k = 0$ and $C = 1$

Example: Not Big-Oh

• Order matters in big-oh. Sometimes $f$ is $O(g)$ and $g$ is $O(f)$, but in general big-oh is not symmetric.

Consider $f(n) = 4n$ and $g(n) = n^2$. $f$ is $O(g)$.

• Can we prove that $g$ is $O(f)$? Formally, there exist constants $C \in \mathbb{N}$ and $k \in \mathbb{R}$ such that $n^2 \leq C \cdot 4n$ whenever $n > k$.

• No. To prove, must show that the negation is true. $\forall C \in \mathbb{N}, \forall k \in \mathbb{R}, \exists n > k$ such that $n^2 > 4Cn$.

$\forall C \in \mathbb{N}, \forall k \in \mathbb{R}, \exists n > k$ such that $n^2 > 4Cn$.

• To prove that the negation is true, start with arbitrary $C$ and $k$. Must show/construct an $n > k$ such that $n^2 > 4Cn$.

• Easy to satisfy $n > k$, then

• To satisfy $n^2 > 4Cn$, divide both sides by $n$ to get $n > 4C$. Pick $n = \max(4C + 1, k)$
Is $2^n \mathcal{O}(n!)$?

We must show that $\exists$ constants $C \in \mathbb{N}$ and $k \in \mathbb{Z}$ such that $|2^n| \leq C|n!|$ whenever $n > k$.

$2^n = 2(2)(2) \ldots (2)(2)(2)$ $n$ times

$\leq n(n-1)(n-2) \ldots (3)(2)(1) = n!$ if $n = 4$

So let $C = 1$ and $k = 3$.

Is $2^n \mathcal{O}(n!)$?

Note that we could also choose $k = 1$ and $C = 2$

Since

$|2^0| \leq 2^0|0!| = 2$

$|2^1| \leq 2^1|1!| = 2$

$|2^2| \leq 2^2|2!| = 4$

$|2^3| \leq 2^3|3!| = 12$

Is $f(x) = (x^2+1)/(x+1) \mathcal{O}(x)$?

We must show that $\exists$ constants $C \in \mathbb{N}$ and $k \in \mathbb{R}$ such that $|f(x)| \leq C|x|$ whenever $x > k$.

$x - 1 + \frac{2}{x+1} \frac{x}{x+1}$

When $x > 1$ (Why?)

Therefore let $k = 1$, $C = 1$

$|f(x)| \leq |x|$ when $x > 1$

Hierarchy of functions

$\frac{1}{n} \log_2 n \ n \ \sqrt{n} \ 3\sqrt{n}$

$1 \ n^2 \ n! \ 2^n \ \log_2 n \ n^n$

$\frac{1}{n} \log_2 n \ n \ \sqrt{n} \ 3\sqrt{n}$

$1, \ \log_2 n \ n^3 \ n^2 \ n! \ 2^n \ n^n \ n^n$
<table>
<thead>
<tr>
<th>Hierarchy of functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1, $\log_2 n$, $\sqrt[3]{n}$, $\sqrt[4]{n}$, $n\sqrt[5]{n}$, $n!$, $2^n$, $n^n$, $n^3$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Hierarchy of functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1, $\log_2 n$, $\sqrt[3]{n}$, $\sqrt[4]{n}$, $n\sqrt[5]{n}$, $n\log_2 n$, $n!$, $2^n$, $n^n$, $n^3$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Hierarchy of functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1, $\log_2 n$, $\sqrt[3]{n}$, $\sqrt[4]{n}$, $n\sqrt[5]{n}$, $n\log_2 n$, $n\sqrt[6]{n}$, $n!$, $2^n$, $n^n$, $n^3$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Hierarchy of functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1, $\log_2 n$, $\sqrt[3]{n}$, $\sqrt[4]{n}$, $n\sqrt[5]{n}$, $n\log_2 n$, $n\sqrt[6]{n}$, $n!$, $2^n$, $n^n$, $n^3$, $n^2$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Hierarchy of functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1, $\log_2 n$, $\sqrt[3]{n}$, $\sqrt[4]{n}$, $n\sqrt[5]{n}$, $n\log_2 n$, $n\sqrt[6]{n}$, $n!$, $2^n$, $n^n$, $n^3$, $n^2$</td>
</tr>
</tbody>
</table>
Hierarchy of functions
1, log₂n, ³√n, √n, n, nlog₂n, n√n, n², n³

1, log₂n, ³√n, √n, n, nlog₂n, n√n, n², n³, 2^n

n! 2^n n^n

Hierarchy of functions
1, log₁₀x is $O(\log_2 x)$
First we will prove the following lemma:
Lemma: $\log_{10} x = c \log_2 x$ where c is a constant.
Proof:
Let $y = \log_2 x$. Then $2^y = x \rightarrow \log_{10} 2^y = \log_{10} x$.
$\log_{10} 2^y = y \log_{10} 2 = \log_{10} x$. But since $y = \log_2 x$, this means that
$\log_2 x \log_{10} 2 = \log_{10} x$. Therefore $c = \log_{10} 2$

To Prove that $\log_{10} x$ is $O(\log_2 x)$
We must show that $\exists$ constants $C \in \mathbb{N}$ and $k \in \mathbb{R}$ such that $|\log_{10} x| \leq C |\log_2 x|$ whenever $x > k$.
From the lemma $\log_2 x \log_{10} 2 = \log_{10} x$ so
choose $C = \log_{10} 2$, $k=1$

Prove log(n!) is $O(n \log n)$
We must show that $\exists$ constants $C \in \mathbb{N}$ and $k \in \mathbb{R}$ such that $|\log(n!)| \leq C |n \log n|$ whenever $x > k$.
We know that $n! \leq n^n$ so
$\log(n!) \leq \log(n^n) = n \log n$
So choose $k = 1$, $C = 1$
Why do we care about growth of functions?

We care about the running time of computer programs!

Running time of a program depends on factors such as:

1. The input to the program
2. The quality of the code generated by the compiler.
3. The nature and speed of the instructions on the machine used to execute the program.
4. The time complexity of the algorithm underlying the program.

**Time Complexity**

- The amount of time required for an algorithm to solve a problem.
- Since different computers run at different speeds we actually describe time complexity in terms of the number of operations required to solve a problem as a function of the size of the input.
- \( T(n) = \text{time complexity of a problem of size } n \).

**Example:** Find the location of a particular element, \( k \), in a sequential list of \( n \) elements.

- **Best case complexity**
  - We find \( k \) in the first location we check.
- **Worst case complexity**
  - We find \( k \) in the last (\( n^{th} \)) location we check.
- **Average case complexity**
  - We find \( k \) after searching \( n/2 \) locations.

**Big-O Notation for Time Complexity**

- We say that the time complexity of a function, \( T(n) \) is \( O(g(n)) \) if there are constants \( C \in \mathbb{N} \) and \( k \in \mathbb{N} \) such that
  \[ |T(n)| \leq C|g(n)| \text{ whenever } n > k. \]

This is exactly the same thing as the growth of functions definition with the assumption that the input is a natural number.