CSC3310 Algorithms

Asymptotic Notation

William Retert

Milwaukee School of Engineering

Example—CONTAINS

Given an array A and a value v, determine whether the value is stored in the array.

Example—CONTAINS

Given an array A and a value v, determine whether the value is stored in the array.

Input An array $A[0, \ldots, n-1]$ of n elements and a value v
Output True iff v is one of the elements of the array; false otherwise

Example—CONTAINS

Given an array A and a value v, determine whether the value is stored in the array.

Input An array $A[0, \ldots, n-1]$ of n elements and a value v
Output True iff v is one of the elements of the array; false otherwise

```
procedure CONTAINS(A[0,...,n-1], v)
    check ← 0
    while check ≠ n and A[check] ≠ v do
        check ← check + 1
    end while
    return check ≠ n
end procedure
```

Example—CONTAINS

```
procedure CONTAINS(A[0,...,n-1], v)
    check ← 0
    while check ≠ n and A[check] ≠ v do
        check ← check + 1
    end while
    return check ≠ n
end procedure
```

We can define the loop invariant as

$$\forall i < check, A[i] \neq v$$

This is trivially true on the first iteration, and inductively true for subsequent iterations. (From the loop test for the current iteration and the induction for the previous ones)
When the loop ends, then either $check = n$ or $A[check] = v$. If $check = n$, then the loop invarient implies that v is not in the array. If $check \neq n$ then $A[check] = v$, so v is in the array.

Example—CONTAINS

```
procedure CONTAINS(A[0,...,n-1], v)
    check ← 0
    while check ≠ n and A[check] ≠ v do
        check ← check + 1
    end while
    return check ≠ n
end procedure
```

Define $T(n)$ as the number of single-step operations peformed on an array of n elements.

$$T(n) = \begin{cases} 3n+3 & v \text{ is not in } A \\ 3(\text{indexof}(v))+4 & v \text{ in } A \end{cases}$$

Example—CONTAINS

Define $T(n)$ as the number of single-step operations peformed on an array of n elements.

$$T(n) = \begin{cases} 3n+3 & v \text{ is not in } A \\ 3(\text{indexof}(v))+4 & v \text{ in } A \end{cases}$$

The time depends on n, but also on whether and where v is in the array. We therefore consider cases:

Best case v is the first thing in the array. We find it in 4 steps
Worst case v is not in the array. We perform $3n+3$ steps looking for it.

N.B. The size of n is not a case! All cases are for values of $T(n)$, where the n is passed in from outside.

Time Complexity

Benchmark Implement and measure

- Hardware?
- Software Environment?

Exact Analysis Count operations

- Laborious
- What is an operation?

Asymptotic Analysis Estimate number of primitive operations within a constant factor

- Relative to size of input
- Ignore fine details of implementation

Asymptotic Notation

Five kinds of notation:

- Θ “Big Theta” — Upper and Lower Bound
- O “Big O” — Inclusive Upper Bound
- Ω “Big Omega” — Inclusive Lower Bound
- o “Little O” — Exclusive Upper Bound
- ω “Little Omega” — Exclusive Lower Bound

Asymptotic Analysis

```
Scanner in = new Scanner(System.in);
String msg = "How many friends do you want to say hello to?";
System.out.println(msg);
int numFriends = Integer.parseInt(in.nextLine());
String s = "Hello to " + numFriends + " friends!";
System.out.println(s);
```

Operations:

Asymptotic Analysis

```
Scanner in = new Scanner(System.in);
String msg = "How many friends do you want to say hello to?";
System.out.println(msg);
int numFriends = Integer.parseInt(in.nextLine());
String s = "Hello to " + numFriends + " friends!";
System.out.println(s);
```

Operations:

3

Asymptotic Analysis

```
Scanner in = new Scanner(System.in);
String msg = "How many friends do you want to say hello to?";
System.out.println(msg);
int numFriends = Integer.parseInt(in.nextLine());
String s = "Hello to " + numFriends + " friends!";
System.out.println(s);
```

Operations:

$$3 + 1$$

Asymptotic Analysis

```
Scanner in = new Scanner(System.in);
String msg = "How many friends do you want to say hello to?";
System.out.println(msg);
int numFriends = Integer.parseInt(in.nextLine());
String s = "Hello to " + numFriends + " friends!";
System.out.println(s);
```

Operations:

$$3 + 1 + 3$$

Asymptotic Analysis

```
Scanner in = new Scanner(System.in);
String msg = "How many friends do you want to say hello to?";
System.out.println(msg);
int numFriends = Integer.parseInt(in.nextLine());
String s = "Hello to " + numFriends + " friends!";
System.out.println(s);
```

Operations:

$$3 + 1 + 3 + 5 + ??$$

Asymptotic Analysis

```
Scanner in = new Scanner(System.in);
String msg = "How many friends do you want to say hello to?";
System.out.println(msg);
int numFriends = Integer.parseInt(in.nextLine());
String s = "Hello to " + numFriends + " friends!";
System.out.println(s);
```

Operations:

$$\cancel{3}O(1) + O(1)\cancel{1} + O(1)\cancel{3} + O(1)$$

Asymptotic Analysis

```
Scanner in = new Scanner(System.in);
String msg = "How many friends do you want to say hello to?";
System.out.println(msg);
int numFriends = Integer.parseInt(in.nextLine());
String s = "Hello to " + numFriends + " friends!";
System.out.println(s);
```

Operations:

$$O(1) + O(1) + O(1) + O(1) + O(1)$$

Asymptotic Analysis

```
Scanner in = new Scanner(System.in);
String msg = "How many friends do you want to say hello to?";
System.out.println(msg);
int numFriends = Integer.parseInt(in.nextLine());
String s = "Hello to " + numFriends + " friends!";
System.out.println(s);
```

Operations:

$$O(1) + O(1) + O(1) + O(1) + O(1) + O(1)$$

Asymptotic Analysis

```
Scanner in = new Scanner(System.in);
String msg = "How many friends do you want to say hello to?";
System.out.println(msg);
int numFriends = Integer.parseInt(in.nextLine());
String s = "Hello to " + numFriends + " friends!";
System.out.println(s);
```

Operations:

$$O(1) + O(1) + O(1) + O(1) + O(1) + O(1) = 6 \times O(1)$$

Asymptotic Analysis

```
Scanner in = new Scanner(System.in);
String msg = "How many friends do you want to say hello to?";
System.out.println(msg);
int numFriends = Integer.parseInt(in.nextLine());
String s = "Hello to " + numFriends + " friends!";
System.out.println(s);
```

Operations:

$$O(1) + O(1) + O(1) + O(1) + O(1) + O(1) = O(1)$$

Loops

```
public static int find(int[] a, int target) {
  int result = -1;
  for(int i = 0; i < a.length; ++i) {
    if( a[i] == target ) {
      result = i;
    }
  }
  return result;
}
```

Loops

```
public static int find(int[] a, int target) {
  int result = -1;
  for(int i = 0; i < a.length; ++i) {
    if( a[i] == target ) {
      result = i;
    }
  }
  return result;
}
```

Operations: $O(1)$

Loops

```
public static int find(int[] a, int target) {
  int result = -1;
  for(int i = 0; i < a.length; ++i) {
    if( a[i] == target ) {
      result = i;
    }
  }
  return result;
}
```

Operations: $O(1) + O(1)$

Loops

```
public static int find(int[] a, int target) {
  int result = -1;
  for(int i = 0; i < a.length; ++i) {
    if( a[i] == target ) {
      result = i;
    }
  }
  return result;
}
```

Operations: $O(1) + O(1) + \sum_{i=0}^{n-1}()$

Loops

```
public static int find(int[] a, int target) {
  int result = -1;
  for(int i = 0; i < a.length; ++i) {
    if( a[i] == target ) {
      result = i;
    }
  }
  return result;
}
```

Operations: $O(1) + O(1) + \sum_{i=0}^{n-1}(O(1))$

Loops

```
public static int find(int[] a, int target) {
  int result = -1;
  for(int i = 0; i < a.length; ++i) {
    if( a[i] == target ) {
      result = i;
    }
  }
  return result;
}
```

Operations: $O(1) + O(1) + \sum_{i=0}^{n-1}(O(1) + O(1))$

Loops

```
public static int find(int[] a, int target) {
  int result = -1;
  for(int i = 0; i < a.length; ++i) {
    if( a[i] == target ) {
      result = i;
    }
  }
  return result;
}
```

Operations: $O(1) + O(1) + \sum_{i=0}^{n-1}(O(1) + O(1) + O(1))$

Loops

```
public static int find(int[] a, int target) {
  int result = -1;
  for(int i = 0; i < a.length; ++i) {
    if( a[i] == target ) {
      result = i;
    }
  }
  return result;
}
```

Operations: $O(1) + O(1) + \sum_{i=0}^{n-1}(O(1) + O(1) + O(1) + O(1))$

Loops

```
public static int find(int[] a, int target) {
  int result = -1;
  for(int i = 0; i < a.length; ++i) {
    if( a[i] == target ) {
      result = i;
    }
  }
  return result;
}
```

Operations: $O(1) + O(1) + \sum_{i=0}^{n-1}(O(1) + O(1) + O(1) + O(1)) + O(1)$

Loops

```
public static int find(int[] a, int target) {
  int result = -1;
  for(int i = 0; i < a.length; ++i) {
    if( a[i] == target ) {
      result = i;
    }
  }
  return result;
}
```

Operations: $O(1) + O(1) + \sum_{i=0}^{n-1}(O(1) + O(1) + O(1) + O(1)) + O(1) + O(1)$

Loops

```
public static int find(int[] a, int target) {
  int result = -1;
  for(int i = 0; i < a.length; ++i) {
    if( a[i] == target ) {
      result = i;
    }
  }
  return result;
}
```

Operations: $O(1) + O(1) + \sum_{i=0}^{n-1}(O(1) + O(1) + O(1) + O(1)) + O(1) + O(1)$

$$= 4nO(1) + 4O(1)$$

Loops

```
public static int find(int[] a, int target) {
  int result = -1;
  for(int i = 0; i < a.length; ++i) {
    if( a[i] == target ) {
      result = i;
    }
  }
  return result;
}
```

Operations: $O(1) + O(1) + \sum_{i=0}^{n-1}(O(1) + O(1) + O(1) + O(1)) + O(1) + O(1)$

$$= 4nO(1) + 4O(1)$$

$$= O(n)$$

Simplifying Big-O

- If $d(n)$ is $O(f(n))$ then $ad(n)$ is $O(f(n))$ for constant $a > 0$
- If $d(n)$ is $O(f(n))$ and $e(n)$ is $O(g(n))$ then $d(n) + e(n)$ is $O(f(n) + g(n))$
- If $d(n)$ is $O(f(n))$ and $e(n)$ is $O(g(n))$ then $d(n)e(n)$ is $O(f(n)g(n))$
- If $d(n)$ is $O(f(n))$ and $f(n)$ is $O(g(n))$ then $d(n)$ is $O(g(n))$
- If $f(n)$ is a polynomial of degree d, then $f(n)$ is $O\left(n^d\right)$

- n^x is $O(a^n)$ for any fixed $x > 0$ and $a > 1$
- $\log(n^x)$ is $O(\log n)$ for any fixed $x > 0$
- $\log^x n$ is $O(n^y)$ for any fixed $x > 0$ and $y > 0$

N.B. Including constant factors and lower-order terms in big-O notation is considered poor taste.

Ranking Families

"Largest" to "smallest":

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

- Functions which are $O(n^k)$ for some $k \geq 1$ are called *polynomial*
- Functions which are $O(a^n)$ for some $a > 1$ are called *exponential*

Ranking Families

"Largest" to "smallest":

$\Theta(n!)$

Exponential $\Theta(2^n)$

$\dots$

$\Theta(n^3)$

$\Theta(n^2 \log n)$

Quadratic $\Theta(n^2)$

$\Theta(n \log n)$

Linear $\Theta(n)$

Logarithmic $\Theta(\log n)$

Constant $\Theta(1)$

- Functions which are $\Theta(n^k)$ for some $k \geq 1$ are called *polynomial*
- Functions which are $\Theta(a^n)$ for some $a > 1$ are called *exponential*

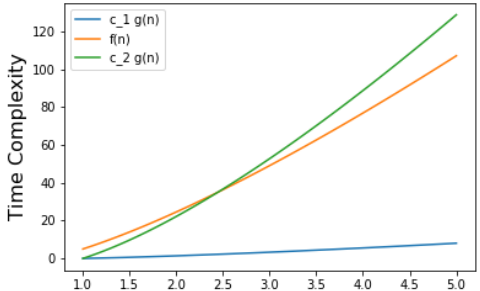
Asymptotic Notation

Five kinds of notation:

- Θ “Big Theta” — Upper and Lower Bound
- O “Big O” — Inclusive Upper Bound
- Ω “Big Omega” — Inclusive Lower Bound
- o “Little O” — Exclusive Upper Bound
- ω “Little Omega” — Exclusive Lower Bound

Θ Notation

- $\Theta(g(n))$ asymptotically-tight bound
  - Upper bound of $c_2 g(n)$ (grows slower than $c_2 g(n)$)
  - Lower bound of $c_1 g(n)$ (grows faster than $c_1 g(n)$)
- We get to choose c_1 and c_2
- Only holds for large enough n



Θ Notation

$$\Theta(g(n)) = \left\{ f(n) \;\middle|\; \begin{array}{l} \text{there exist positive constants } c_1, c_2, \text{ and } n_0 \\ \text{such that } 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \text{ for all } n \geq n_0 \end{array} \right\}$$

Θ Notation

$$\Theta(g(n)) = \left\{ f(n) \;\middle|\; \begin{array}{l} \text{there exist positive constants } c_1, c_2, \text{ and } n_0 \\ \text{such that } 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \text{ for all } n \geq n_0 \end{array} \right\}$$

$\{x|condition\}$ is the set of **all** x's for which *condition* is true.

- x is in the set **iff** the condition is true
  - if *condition* is true for x, then x is in the set
  - if x is in the set, then *condition* is true for x

Θ Notation

$$\Theta(g(n)) = \left\{ f(n) \;\middle|\; \begin{array}{l} \text{there exist positive constants } c_1, c_2, \text{ and } n_0 \\ \text{such that } 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \text{ for all } n \geq n_0 \end{array} \right\}$$

$\{x|condition\}$ is the set of **all** x's for which *condition* is true.

- x is in the set **if and only if** the condition is true
  - if *condition* is true for x, then x is in the set
  - if x is in the set, then *condition* is true for x

Θ Notation

$$\Theta(g(n)) = \left\{ f(n) \;\middle|\; \begin{array}{l} \text{there exist positive constants } c_1, c_2, \text{ and } n_0 \\ \text{such that } 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \text{ for all } n \geq n_0 \end{array} \right\}$$

Here, $f(n) \in \Theta(g(n))$ if and only if $0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)$ for $n \geq n_0$
That is, $\Theta(g(n))$ defines a set of functions.

- Some functions are in the set, some are not.
- We can find c_1, c_2, and n_0 for any function in the set.

Θ Notation

$$\Theta(g(n)) = \left\{ f(n) \;\middle|\; \begin{array}{l} \text{there exist positive constants } c_1, c_2, \text{ and } n_0 \\ \text{such that } 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \text{ for all } n \geq n_0 \end{array} \right\}$$

Here, $f(n) \in \Theta(g(n))$ if and only if $0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)$ for $n \geq n_0$
Let $f(n) = 5n^2 - 3n + 2$. We can show that $f(n) \in \Theta\left(n^2\right)$

Θ Notation

$$\Theta(g(n)) = \left\{ f(n) \;\middle|\; \begin{array}{l} \text{there exist positive constants } c_1, c_2, \text{ and } n_0 \\ \text{such that } 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \text{ for all } n \geq n_0 \end{array} \right\}$$

Here, $f(n) \in \Theta(g(n))$ if and only if $0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)$ for $n \geq n_0$
Let $f(n) = 5n^2 - 3n + 2$. We can show that $f(n) \in \Theta\left(n^2\right)$ To do this, we must show that

$$c_1 n^2 \leq 5n^2 - 3n + 2 \leq c_2 n^2$$

Θ Notation

$$\Theta(g(n)) = \left\{ f(n) \;\middle|\; \begin{array}{l} \text{there exist positive constants } c_1, c_2, \text{ and } n_0 \\ \text{such that } 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \text{ for all } n \geq n_0 \end{array} \right\}$$

Here, $f(n) \in \Theta(g(n))$ if and only if $0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)$ for $n \geq n_0$
Let $f(n) = 5n^2 - 3n + 2$. We can show that $f(n) \in \Theta\left(n^2\right)$ To do this, we must show that

$$c_1 n^2 \leq 5n^2 - 3n + 2 \leq c_2 n^2$$

Let's split that up:

$c_1 n^2 \leq 5n^2 - 3n + 2$
$5n^2 - 3n + 2 \leq c_2 n^2$

Θ Notation

$$\Theta(g(n)) = \left\{ f(n) \mid \begin{array}{l} \text{there exist positive constants } c_1, c_2, \text{ and } n_0 \\ \text{such that } 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \text{ for all } n \geq n_0 \end{array} \right\}$$

Let $f(n) = 5n^2 - 3n + 2$. We can show that $f(n) \in \Theta\left(n^2\right)$ To do this, we must show that

$$c_1 n^2 \leq 5n^2 - 3n + 2 \leq c_2 n^2$$

Let's split that up:

$c_1 n^2 \leq 5n^2 - 3n + 2$ This is true for $c_1 = 4$ if $n \geq 2$

$$4n^2 \leq 5n^2 - 3n + 2$$
$$3n \leq n^2 + 2$$

$5n^2 - 3n + 2 \leq c_2 n^2$

Θ Notation

$$\Theta(g(n)) = \left\{ f(n) \mid \begin{array}{l} \text{there exist positive constants } c_1, c_2, \text{ and } n_0 \\ \text{such that } 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \text{ for all } n \geq n_0 \end{array} \right\}$$

Let $f(n) = 5n^2 - 3n + 2$. We can show that $f(n) \in \Theta\left(n^2\right)$ To do this, we must show that

$$c_1 n^2 \leq 5n^2 - 3n + 2 \leq c_2 n^2$$

Let's split that up:

$c_1 n^2 \leq 5n^2 - 3n + 2$ This is true for $c_1 = 4$ if $n \geq 2$

$$\begin{aligned} 4n^2 &\leq 5n^2 - 3n + 2 \\ 3n &\leq n^2 + 2 \end{aligned}$$

$5n^2 - 3n + 2 \leq c_2 n^2$ This is true for $c_2 = 5$

$$\begin{aligned} 5n^2 - 3n + 2 &\leq 5n^2 \\ 2 &\leq 3n \end{aligned}$$

Θ Notation

$$\Theta(g(n)) = \left\{ f(n) \mid \begin{array}{l} \text{there exist positive constants } c_1, c_2, \text{ and } n_0 \\ \text{such that } 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \text{ for all } n \geq n_0 \end{array} \right\}$$

Let $f(n) = 5n^2 - 3n + 2$. We can show that $f(n) \in \Theta\left(n^2\right)$ To do this, we must show that

$$c_1 n^2 \leq 5n^2 - 3n + 2 \leq c_2 n^2$$

Let's split that up:

$c_1 n^2 \leq 5n^2 - 3n + 2$ This is true for $c_1 = 4$ if $n \geq 2$

$$\begin{aligned} 4n^2 &\leq 5n^2 - 3n + 2 \\ 3n &\leq n^2 + 2 \end{aligned}$$

$5n^2 - 3n + 2 \leq c_2 n^2$ This is true for $c_2 = 5$

$$\begin{aligned} 5n^2 - 3n + 2 &\leq 5n^2 \\ 2 &\leq 3n \end{aligned}$$

unless $n = 0 \ldots$

Θ Notation

$$\Theta(g(n)) = \left\{ f(n) \mid \begin{array}{l} \text{there exist positive constants } c_1, c_2, \text{ and } n_0 \\ \text{such that } 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \text{ for all } n \geq n_0 \end{array} \right\}$$

Let $f(n) = 5n^2 - 3n + 2$. We can show that $f(n) \in \Theta\left(n^2\right)$ To do this, we must show that

$$c_1 n^2 \leq 5n^2 - 3n + 2 \leq c_2 n^2$$

Let's split that up:

$c_1 n^2 \leq 5n^2 - 3n + 2$ This is true for $c_1 = 4$ if $n \geq 2$

$$\begin{aligned} 4n^2 &\leq 5n^2 - 3n + 2 \\ 3n &\leq n^2 + 2 \end{aligned}$$

$5n^2 - 3n + 2 \leq c_2 n^2$ This is true for $c_2 = 5$ and $n \geq 2$

$$\begin{aligned} 5n^2 - 3n + 2 &\leq 5n^2 \\ 2 &\leq 3n \end{aligned}$$

Θ Notation

$$\Theta(g(n)) = \left\{ f(n) \mid \begin{array}{l} \text{there exist positive constants } c_1, c_2, \text{ and } n_0 \\ \text{such that } 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \text{ for all } n \geq n_0 \end{array} \right\}$$

Let $f(n) = 5n^2 - 3n + 2$. We can show that $f(n) \in \Theta\left(n^2\right)$ To do this, we must show that

$$c_1 n^2 \leq 5n^2 - 3n + 2 \leq c_2 n^2$$

Let's split that up:

$c_1 n^2 \leq 5n^2 - 3n + 2$ This is true for $c_1 = 4$ if $n \geq 2$

$$4n^2 \leq 5n^2 - 3n + 2$$
$$3n \leq n^2 + 2$$

$5n^2 - 3n + 2 \leq c_2 n^2$ This is true for $c_2 = 5$ and $n \geq 2$

$$5n^2 - 3n + 2 \leq 5n^2$$
$$2 \leq 3n$$

So we can choose $c_1 = 4$, $c_2 = 5$, and $n_0 = 3$ to meet the condition. Therefore, $5n^2 - 3n + 2 \in \Theta\left(n^2\right)$

Θ Notation

$$\Theta(g(n)) = \left\{ f(n) \mid \begin{array}{l} \text{there exist positive constants } c_1, c_2, \text{ and } n_0 \\ \text{such that } 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \text{ for all } n \geq n_0 \end{array} \right\}$$

Let $f(n) = 5n^2 - 3n + 2$. We can show that $f(n) \in \Theta\left(n^2\right)$ To do this, we must show that

$$c_1 n^2 \leq 5n^2 - 3n + 2 \leq c_2 n^2$$

Let's split that up:

$c_1 n^2 \leq 5n^2 - 3n + 2$ This is true for $c_1 = 1$

$$1n^2 \leq 5n^2 - 3n + 2$$
$$3n \leq 4n^2 + 2$$

$5n^2 - 3n + 2 \leq c_2 n^2$ This is true for $c_2 = 12$ and $n \geq 40$

$$5n^2 - 3n + 2 \leq 12n^2$$
$$2 \leq 7n^2 + 3n$$

So we can choose $c_1 = 4$, $c_2 = 5$, and $n_0 = 3$ to meet the condition. We can also choose $c_1 = 1$, $c_2 = 12$ and $n_0 = 40$. Therefore, $5n^2 - 3n + 2 \in \Theta\left(n^2\right)$

Example Proof: $5n + 2$ is $\Theta(n)$

Example Proof: $5n + 2$ is $\Theta(n)$

By definition $5n + 2 \in \Theta(n)$ if $c_1 n \leq 5n + 2 \leq c_2 n$ for $n \geq n_0$.

Example Proof: $5n + 2$ is $\Theta(n)$

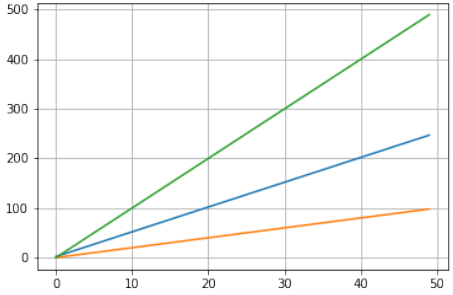
By definition $5n + 2 \in \Theta(n)$ if $c_1 n \leq 5n + 2 \leq c_2 n$ for $n \geq n_0$.
Let $c_1 = 2$, $c_2 = 10$, and $n_0 = 10$

Example Proof: $5n + 2$ is $\Theta(n)$

By definition $5n + 2 \in \Theta(n)$ if $c_1 n \leq 5n + 2 \leq c_2 n$ for $n \geq n_0$.
Let $c_1 = 2$, $c_2 = 10$, and $n_0 = 10$
Is $2n \leq 5n + 2 \leq 10n$ true for $n \geq 10$?

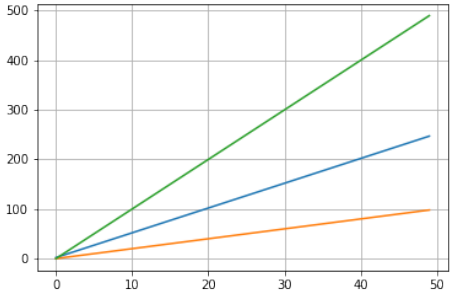
Example Proof: $5n + 2$ is $\Theta(n)$

By definition $5n + 2 \in \Theta(n)$ if $c_1 n \leq 5n + 2 \leq c_2 n$ for $n \geq n_0$.
Let $c_1 = 2$, $c_2 = 10$, and $n_0 = 10$
Is $2n \leq 5n + 2 \leq 10n$ true for $n \geq 10$?



Example Proof: $5n + 2$ is $\Theta(n)$

By definition $5n + 2 \in \Theta(n)$ if $c_1 n \leq 5n + 2 \leq c_2 n$ for $n \geq n_0$.
Let $c_1 = 2$, $c_2 = 10$, and $n_0 = 10$
As $2n \leq 5n + 2 \leq 10n$ for $n \geq 10$, $5n + 2 \in \Theta(n)$



Example Proof: $5n + 2 \notin \Theta\left(n^2\right)$

Example Proof: $5n + 2 \notin \Theta\left(n^2\right)$

Proof by contradiction: Assume the opposite and derive a contradiction.

Example Proof: $5n + 2 \notin \Theta\left(n^2\right)$

Proof by contradiction: Assume that $5n + 2 \in \Theta\left(n^2\right)$

Example Proof: $5n + 2 \notin \Theta\left(n^2\right)$

Proof by contradiction: Assume that $5n + 2 \in \Theta\left(n^2\right)$
Then, by the definition, $c_1 n^2 \leq 5n + 2 \leq c_2 n^2$ when $n \geq n_0$ for some c_1, c_2, n_0. That is:

- $c_1 n^2 \leq 5n + 2$
- $5n + 2 \leq c_2 n^2$

Which of these is more likely to lead to a contradiction?

Example Proof: $5n + 2 \notin \Theta\left(n^2\right)$

Proof by contradiction: Assume that $5n + 2 \in \Theta\left(n^2\right)$
Then, by the definition, $c_1 n^2 \leq 5n + 2 \leq c_2 n^2$ when $n \geq n_0$ for some c_1, c_2, n_0. That is:

- $c_1 n^2 \leq 5n + 2$
- $5n + 2 \leq c_2 n^2$ True for $c_2 = 5, n \geq 5$

Example Proof: $5n + 2 \notin \Theta\left(n^2\right)$

Proof by contradiction: Assume that $5n + 2 \in \Theta\left(n^2\right)$
Then, by the definition, $c_1 n^2 \leq 5n + 2 \leq c_2 n^2$ when $n \geq n_0$ for some c_1, c_2, n_0. That is:

- $c_1 n^2 \leq 5n + 2$
  Suppose this is true for $c_1 = k$
- $5n + 2 \leq c_2 n^2$

Example Proof: $5n + 2 \notin \Theta\left(n^2\right)$

Proof by contradiction: Assume that $5n + 2 \in \Theta\left(n^2\right)$
Then, by the definition, $c_1 n^2 \leq 5n + 2 \leq c_2 n^2$ when $n \geq n_0$ for some c_1, c_2, n_0. That is:

- $c_1 n^2 \leq 5n + 2$
  Suppose this is true for $c_1 = k$

$$kn^2 \leq 5n + 2 \text{ for all } n \geq n_0$$

- $5n + 2 \leq c_2 n^2$

Example Proof: $5n + 2 \notin \Theta\left(n^2\right)$

Proof by contradiction: Assume that $5n + 2 \in \Theta\left(n^2\right)$
Then, by the definition, $c_1 n^2 \leq 5n + 2 \leq c_2 n^2$ when $n \geq n_0$ for some c_1, c_2, n_0. That is:

- $c_1 n^2 \leq 5n + 2$
  Suppose this is true for $c_1 = k$

$$kn^2 \leq 5n + 2 \text{ for all } n \geq n_0$$
$$n \leq \frac{5}{k} + \frac{2}{kn} \text{ for all } n \geq n_0$$

Example Proof: $5n + 2 \notin \Theta\left(n^2\right)$

Proof by contradiction: Assume that $5n + 2 \in \Theta\left(n^2\right)$
Then, by the definition, $c_1 n^2 \leq 5n + 2 \leq c_2 n^2$ when $n \geq n_0$ for some c_1, c_2, n_0. That is:

- $c_1 n^2 \leq 5n + 2$
  Suppose this is true for $c_1 = k$

$$kn^2 \leq 5n + 2 \text{ for all } n \geq n_0$$
$$n \leq \frac{5}{k} + \frac{2}{kn} \text{ for all } n \geq n_0$$

Let $n = \max(\frac{5}{k} + \frac{2}{kn_0}, n_0 + 1)$.

Example Proof: $5n + 2 \notin \Theta\left(n^2\right)$

Proof by contradiction: Assume that $5n + 2 \in \Theta\left(n^2\right)$

Then, by the definition, $c_1 n^2 \leq 5n + 2 \leq c_2 n^2$ when $n \geq n_0$ for some c_1, c_2, n_0. That is:

- $c_1 n^2 \leq 5n + 2$
  Suppose this is true for $c_1 = k$

$$kn^2 \leq 5n + 2 \text{ for all } n \geq n_0$$

$$n \leq \frac{5}{k} + \frac{2}{kn} \text{ for all } n \geq n_0$$

Let $n = \max(\frac{5}{k} + \frac{2}{kn_0}, n_0 + 1)$. Therefore, $n \geq \frac{5}{k} + \frac{2}{kn_0}$ and $n > n_0$. As $n > n_0$,

$$\frac{5}{k} + \frac{2}{kn_0} \leq n \leq \frac{5}{k} + \frac{2}{kn}$$

$$\cancel{\frac{5}{k}} + \frac{2}{kn_0} \leq \cancel{\frac{5}{k}} + \frac{2}{kn}$$

$$\frac{\cancel{2}}{\cancel{k}n_0} \frac{\cancel{k}n\cancel{n_0}}{\cancel{2}} \leq \frac{\cancel{2}}{\cancel{k}\cancel{n}} \frac{\cancel{k}\cancel{n}n_0}{\cancel{2}}$$

$$n \leq n_0$$

which is a contradiction

Example Proof: $5n + 2 \notin \Theta\left(n^2\right)$

Proof by contradiction: Assume that $5n + 2 \in \Theta\left(n^2\right)$
Then, by the definition, $c_1 n^2 \leq 5n + 2 \leq c_2 n^2$ when $n \geq n_0$ for some c_1, c_2, n_0. That is:

- $c_1 n^2 \leq 5n + 2$
  Suppose this is true for $c_1 = k$

$$kn^2 \leq 5n + 2 \text{ for all } n \geq n_0$$

$$n \leq \frac{5}{k} + \frac{2}{kn} \text{ for all } n \geq n_0$$

$$\infty = \lim_{n \to \infty} n$$

$$\frac{5}{k} = \lim_{n \to \infty} \frac{5}{k} + \frac{2}{kn}$$

Simplifying Big-Θ

- If $d(n)$ is $\Theta(f(n))$ then $ad(n)$ is $\Theta(f(n))$ for constant $a > 0$
- If $d(n)$ is $\Theta(f(n))$ and $e(n)$ is $\Theta(g(n))$ then $d(n) + e(n)$ is $\Theta(f(n) + g(n))$
- If $d(n)$ is $\Theta(f(n))$ and $e(n)$ is $\Theta(g(n))$ then $d(n)e(n)$ is $\Theta(f(n)g(n))$
- If $d(n)$ is $\Theta(f(n))$ and $f(n)$ is $\Theta(g(n))$ then $d(n)$ is $\Theta(g(n))$
- If $f(n)$ is a polynomial of degree d, then $f(n)$ is $\Theta\left(n^d\right)$

- If $f(n) \in \Theta(g(n))$, then $g(n) \in \Theta(f(n))$

N.B. Including constant factors and lower-order terms in big-Θ notation is considered poor taste.

Asymptotic Notation

Five kinds of notation:

- Θ “Big Theta” — Upper and Lower Bound
- O “Big O” — Inclusive Upper Bound
- Ω “Big Omega” — Inclusive Lower Bound
- o “Little O” — Exclusive Upper Bound
- ω “Little Omega” — Exclusive Lower Bound

Big-O, Formally

$$\mathrm{O}\left(g(n)\right) = \left\{ f(n) \;\middle|\; \begin{array}{l} \text{there exist positive constants } c, \text{ and } n_0 \\ \text{such that } 0 \leq f(n) \leq cg(n) \text{ for all } n \geq n_0 \end{array} \right\}$$

Big-O, Formally

$$\mathrm{O}\left(g(n)\right) = \left\{ f(n) \;\middle|\; \begin{array}{l} \text{there exist positive constants } c, \text{ and } n_0 \\ \text{such that } 0 \leq f(n) \leq cg(n) \text{ for all } n \geq n_0 \end{array} \right\}$$

This should look familiar...

$$\Theta\left(g(n)\right) = \left\{ f(n) \;\middle|\; \begin{array}{l} \text{there exist positive constants } c_1, c_2, \text{ and } n_0 \\ \text{such that } 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \text{ for all } n \geq n_0 \end{array} \right\}$$

- As with Θ, O defines a set of functions.
- A function is in the set if we can find appropriate c and n_0

Big-O, Formally

$$\mathrm{O}\left(g(n)\right) = \left\{ f(n) \;\middle|\; \begin{array}{l} \text{there exist positive constants } c, \text{ and } n_0 \\ \text{such that } 0 \leq f(n) \leq cg(n) \text{ for all } n \geq n_0 \end{array} \right\}$$

- As with Θ, O defines a set of functions.
- A function is in the set if we can find appropriate c and n_0

$$5n^2 - 3n + 2 \in \mathrm{O}\left(n^2\right)$$

because we can find $c = 5$ and $n_0 = 2$:

Big-O, Formally

$$\mathrm{O}\left(g(n)\right) = \left\{ f(n) \;\middle|\; \begin{array}{l} \text{there exist positive constants } c, \text{ and } n_0 \\ \text{such that } 0 \leq f(n) \leq cg(n) \text{ for all } n \geq n_0 \end{array} \right\}$$

- As with Θ, O defines a set of functions.
- A function is in the set if we can find appropriate c and n_0

$$5n^2 - 3n + 2 \in \mathrm{O}\left(n^2\right)$$

because we can find $c = 5$ and $n_0 = 2$:

$$5n^2 - 3n + 2 \leq 5n^2$$

$$\cancel{5n^2} - 3n + 2 \leq \cancel{5n^2}$$

$$2 \leq 3n \text{ for all } n \geq 2$$

Big-O Examples

$$64 \in O(1)$$
$$\log n - \log\log n + 2 \in O(\log n)$$
$$\log n - \log\log n + 2 \in O(n)$$
$$5n + \log n \in O(n)$$
$$5n^2 + n\log n \in O\left(n^2\right)$$
$$5n^2 + n\log n \in O\left(n^3\right)$$

Big-O Examples

$$64 \in O(1)$$
$$\log n - \log\log n + 2 \in O(\log n)$$
$$\log n - \log\log n + 2 \in O(n)$$
$$5n + \log n \in O(n)$$
$$5n^2 + n\log n \in O\left(n^2\right)$$
$$5n^2 + n\log n \in O\left(n^3\right)$$
$$5n^2 + n\log n \in O\left(n^4\right)$$
$$5n^2 + n\log n \in O\left(n^5\right)$$
$$5n^2 + n\log n \in O\left(2^n\right)$$

Big-O Examples

$$
\begin{aligned}
64 &\in O(1) \\
\log n - \log\log n + 2 &\in O(\log n) \\
\log n - \log\log n + 2 &\in O(n) \\
5n + \log n &\in O(n) \\
5n^2 + n\log n &\in O\left(n^2\right) \\
5n^2 + n\log n &\in O\left(n^3\right) \\
5n^2 + n\log n &\in O\left(n^4\right) \\
5n^2 + n\log n &\in O\left(n^5\right) \\
5n^2 + n\log n &\in O\left(2^n\right)
\end{aligned}
$$

$$O(1) \subseteq O(\log n) \subseteq O(n) \subseteq O(n\log n) \subseteq O\left(n^2\right) \subseteq O\left(n^2\log n\right) \subseteq O\left(n^3\right) \subseteq O\left(2^n\right) \subseteq O(n!)$$

Big-O Examples

$$5n^2 + n\log n \in O\left(n^2\right)$$
$$5n^2 + n\log n \in O\left(n^3\right)$$
$$5n^2 + n\log n \in O\left(n^4\right)$$
$$5n^2 + n\log n \in O\left(n^5\right)$$
$$5n^2 + n\log n \in O\left(2^n\right)$$

$$O(1) \subseteq O(\log n) \subseteq O(n) \subseteq O(n\log n) \subseteq O\left(n^2\right) \subseteq O\left(n^2\log n\right) \subseteq O\left(n^3\right) \subseteq O\left(2^n\right) \subseteq O(n!)$$

- Conventionally, use the *tightest* ("smallest") bound
  - Provides the most useful information

Prove $\frac{1}{2}n^2 - 3n \in O\left(n^2\right)$

Prove $\frac{1}{2}n^2 - 3n \in O\left(n^2\right)$

Definition:

$$O\left(g(n)\right) = \left\{ f(n) \;\middle|\; \begin{array}{l} \text{there exist positive constants } c, \text{ and } n_0 \\ \text{such that } 0 \leq f(n) \leq cg(n) \text{ for all } n \geq n_0 \end{array} \right\}$$

Prove $\frac{1}{2}n^2 - 3n \in O\left(n^2\right)$

Definition:

$$O\left(g(n)\right) = \left\{ f(n) \;\middle|\; \begin{array}{l} \text{there exist positive constants } c, \text{ and } n_0 \\ \text{such that } 0 \leq f(n) \leq cg(n) \text{ for all } n \geq n_0 \end{array} \right\}$$

Need to find c, n_0 such that $0 \leq \frac{1}{2}n^2 - 3n \leq cn^2$ for all $n \geq n_0$

Prove $\frac{1}{2}n^2 - 3n \in O\left(n^2\right)$

Definition:

$$O\left(g(n)\right) = \left\{ f(n) \;\middle|\; \begin{array}{l} \text{there exist positive constants } c, \text{ and } n_0 \\ \text{such that } 0 \leq f(n) \leq cg(n) \text{ for all } n \geq n_0 \end{array} \right\}$$

Need to find c, n_0 such that $0 \leq \frac{1}{2}n^2 - 3n \leq cn^2$ for all $n \geq n_0$

$$0 \leq \frac{1}{2}n^2 - 3n \leq cn^2$$
$$0 \leq n^2 - 6n \leq 2cn^2$$

Prove $\frac{1}{2}n^2 - 3n \in O\left(n^2\right)$

Definition:

$$O\left(g(n)\right) = \left\{ f(n) \;\middle|\; \begin{array}{l} \text{there exist positive constants } c, \text{ and } n_0 \\ \text{such that } 0 \leq f(n) \leq cg(n) \text{ for all } n \geq n_0 \end{array} \right\}$$

Need to find c, n_0 such that $0 \leq \frac{1}{2}n^2 - 3n \leq cn^2$ for all $n \geq n_0$

$$\begin{aligned} 0 \leq \frac{1}{2}n^2 - 3n &\leq cn^2 \\ 0 \leq n^2 - 6n &\leq 2cn^2 \end{aligned}$$

Let $n_0 = 10$ and $c = 6$

Asymptotic Notation

Five kinds of notation:

- Θ “Big Theta” — Upper and Lower Bound
- O “Big O” — Inclusive Upper Bound
- Ω “Big Omega” — Inclusive Lower Bound
- o “Little O” — Exclusive Upper Bound
- ω “Little Omega” — Exclusive Lower Bound

Big-Ω, Formally

$$\Omega\left(g(n)\right) = \left\{ f(n) \;\middle|\; \begin{array}{l} \text{there exist positive constants } c, \text{ and } n_0 \\ \text{such that } 0 \leq cg(n) \leq f(n) \text{ for all } n \geq n_0 \end{array} \right\}$$

Big-Ω, Formally

$$\Omega\left(g(n)\right) = \left\{ f(n) \;\middle|\; \begin{array}{l} \text{there exist positive constants } c, \text{ and } n_0 \\ \text{such that } 0 \leq cg(n) \leq f(n) \text{ for all } n \geq n_0 \end{array} \right\}$$

$$\Theta\left(g(n)\right) = \left\{ f(n) \;\middle|\; \begin{array}{l} \text{there exist positive constants } c_1, c_2, \text{ and } n_0 \\ \text{such that } 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \text{ for all } n \geq n_0 \end{array} \right\}$$

- As with Θ, Ω defines a set of functions.
- A function is in the set if we can find appropriate c and n_0

Big-Ω, Formally

$$\Omega\left(g(n)\right) = \left\{ f(n) \;\middle|\; \begin{array}{l} \text{there exist positive constants } c, \text{ and } n_0 \\ \text{such that } 0 \leq cg(n) \leq f(n) \text{ for all } n \geq n_0 \end{array} \right\}$$

- As with Θ, Ω defines a set of functions.
- A function is in the set if we can find appropriate c and n_0

$$5n^2 - 3n + 2 \in \Omega\left(n^2\right)$$

because we can find $c = \frac{1}{5}$ and $n_0 = 2$

Prove $\frac{1}{2}n^2 - 3n \in \Omega\left(n^2\right)$

Prove $\frac{1}{2}n^2 - 3n \in \Omega\left(n^2\right)$

Definition:

$$\Omega\left(g(n)\right) = \left\{ f(n) \;\middle|\; \begin{array}{l} \text{there exist positive constants } c, \text{ and } n_0 \\ \text{such that } 0 \leq cg(n) \leq f(n) \text{ for all } n \geq n_0 \end{array} \right\}$$

Prove $\frac{1}{2}n^2 - 3n \in \Omega\left(n^2\right)$

Definition:

$$\Omega\left(g(n)\right) = \left\{ f(n) \;\middle|\; \begin{array}{l} \text{there exist positive constants } c, \text{ and } n_0 \\ \text{such that } 0 \leq cg(n) \leq f(n) \text{ for all } n \geq n_0 \end{array} \right\}$$

Need to find c, n_0 such that $0 \leq cn^2 \leq \frac{1}{2}n^2 - 3n$ for all $n \geq n_0$

Prove $\frac{1}{2}n^2 - 3n \in \Omega\left(n^2\right)$

Definition:

$$\Omega\left(g(n)\right) = \left\{ f(n) \;\middle|\; \begin{array}{l} \text{there exist positive constants } c, \text{ and } n_0 \\ \text{such that } 0 \leq cg(n) \leq f(n) \text{ for all } n \geq n_0 \end{array} \right\}$$

Need to find c, n_0 such that $0 \leq cn^2 \leq \frac{1}{2}n^2 - 3n$ for all $n \geq n_0$

$$0 \leq cn^2 \leq \frac{1}{2}n^2 - 3n$$
$$0 \leq 2cn^2 \leq n^2 - 6n$$

Prove $\frac{1}{2}n^2 - 3n \in \Omega\left(n^2\right)$

Definition:

$$\Omega\left(g(n)\right) = \left\{ f(n) \;\middle|\; \begin{array}{l} \text{there exist positive constants } c, \text{ and } n_0 \\ \text{such that } 0 \leq cg(n) \leq f(n) \text{ for all } n \geq n_0 \end{array} \right\}$$

Need to find c, n_0 such that $0 \leq cn^2 \leq \frac{1}{2}n^2 - 3n$ for all $n \geq n_0$

$$0 \leq cn^2 \leq \frac{1}{2}n^2 - 3n$$
$$0 \leq 2cn^2 \leq n^2 - 6n$$

Let $n_0 = 14$ and $c = \frac{1}{4}$
$6n < \frac{1}{2}n^2$ for $n \geq 14$. Therefore, $n^2 - 6n \geq \frac{1}{2}n^2$

Big-Ω Examples

$$64 \in \Omega(1)$$
$$\log n - \log\log n + 2 \in \Omega(\log n)$$
$$\log n - \log\log n + 2 \in \Omega(1)$$
$$5n + \log n \in \Omega(n)$$
$$5n^2 + n\log n \in \Omega\left(n^2\right)$$
$$5n^2 + n\log n \in \Omega(n\log n)$$

Big-Ω Examples

$$64 \in \Omega(1)$$
$$\log n - \log\log n + 2 \in \Omega(\log n)$$
$$\log n - \log\log n + 2 \in \Omega(1)$$
$$5n + \log n \in \Omega(n)$$
$$5n^2 + n\log n \in \Omega\left(n^2\right)$$
$$5n^2 + n\log n \in \Omega(n\log n)$$
$$5n^2 + n\log n \in \Omega(n)$$
$$5n^2 + n\log n \in \Omega(\log n)$$
$$5n^2 + n\log n \in \Omega(1)$$

Big-Ω Examples

$$64 \in \Omega(1)$$
$$\log n - \log\log n + 2 \in \Omega(\log n)$$
$$\log n - \log\log n + 2 \in \Omega(1)$$
$$5n + \log n \in \Omega(n)$$
$$5n^2 + n\log n \in \Omega\left(n^2\right)$$
$$5n^2 + n\log n \in \Omega(n\log n)$$
$$5n^2 + n\log n \in \Omega(n)$$
$$5n^2 + n\log n \in \Omega(\log n)$$
$$5n^2 + n\log n \in \Omega(1)$$

$$\Omega(n!) \subset \Omega(2^n) \subset \Omega\left(n^3\right) \subset \Omega\left(n^2\log n\right) \subset \Omega\left(n^2\right) \subset \Omega(n\log n) \subset \Omega(n) \subset \Omega(\log n) \subset \Omega(1)$$

Big-Ω Examples

$$5n^2 + n\log n \in \Omega\left(n^2\right)$$
$$5n^2 + n\log n \in \Omega\left(n\log n\right)$$
$$5n^2 + n\log n \in \Omega\left(n\right)$$
$$5n^2 + n\log n \in \Omega\left(\log n\right)$$
$$5n^2 + n\log n \in \Omega\left(1\right)$$

$$\Omega\left(n!\right) \subset \Omega\left(2^n\right) \subset \Omega\left(n^3\right) \subset \Omega\left(n^2\log n\right) \subset \Omega\left(n^2\right) \subset \Omega\left(n\log n\right) \subset \Omega\left(n\right) \subset \Omega\left(\log n\right) \subset \Omega\left(1\right)$$

- Conventionally, use the *tightest* ("largest") bound
  - Provides the most useful information

Relationship between Θ, O and Ω

Theorem

For any two functions $f(n)$ and $g(n)$ $f(n) \in \Theta(g(n))$ if and only if $f(n) \in O(g(n))$ and $f(n) \in \Omega(g(n))$

Relationship between Θ, O and Ω

Theorem

For any two functions $f(n)$ and $g(n)$ $f(n) \in \Theta(g(n))$ if and only if $f(n) \in O(g(n))$ and $f(n) \in \Omega(g(n))$

$$\Theta(g(n)) = \left\{ f(n) \;\middle|\; \begin{array}{l} \text{there exist positive constants } c_1, c_2, \text{ and } n_0 \\ \text{such that } 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \text{ for all } n \geq n_0 \end{array} \right\}$$

$$O(g(n)) = \left\{ f(n) \;\middle|\; \begin{array}{l} \text{there exist positive constants } c, \text{ and } n_0 \\ \text{such that } 0 \leq f(n) \leq cg(n) \text{ for all } n \geq n_0 \end{array} \right\}$$

$$\Omega(g(n)) = \left\{ f(n) \;\middle|\; \begin{array}{l} \text{there exist positive constants } c, \text{ and } n_0 \\ \text{such that } 0 \leq cg(n) \leq f(n) \text{ for all } n \geq n_0 \end{array} \right\}$$