CSE 250: Asymptotic Analysis

Lecture 7

Sept 11, 2024

Reminders

- PA1 Tests due Sun, Sept 15 at 11:59 PM
  - Recitations will cover writing good test cases.
- PA1 Implementation due Sun, Sept 22 at 11:59 PM
  - Implement a Sorted Linked List

Examples

$$n^2 + 4n \stackrel{?}{\in} \theta(n^2)$$

Examples

$$n^2 + 4n \stackrel{?}{\in} \theta(n^2)$$

$$2^n + 4n \stackrel{?}{\in} \theta(n^2)$$

Examples

$$n^2 + 4n \stackrel{?}{\in} \theta(n^2)$$

$$2^n + 4n \stackrel{?}{\in} \theta(n^2)$$

Shortcut: Find the dominant term being summed, and compare it.

Tight Bounds

If $g(N) \in \theta(f(N))$:

- $g(N) \in O(f(N))$ is a **tight bound**.
- $g(N) \in \Omega(f(N))$ is a **tight bound**.

Examples

```
1  public void updateUsers(User[] users)
2  {
3    x = 1;
4    for(user : users)
5    {
6      user.id = x;
7    }
8  }
```

$$1 + \sum_{\texttt{user} \in \texttt{users}} 2 \text{ steps} =$$

Examples

```
1  public void updateUsers(User[] users)
2  {
3    x = 1;
4    for(user : users)
5    {
6      user.id = x;
7    }
8  }
```

$$1 + \sum_{\texttt{user} \in \texttt{users}} 2 \text{ steps} = 1 + 2 \times |\texttt{users}|$$

Examples

```
1  public void updateUsers(User[] users)
2  {
3    x = 1;
4    for(user : users)
5    {
6      user.id = x;
7    }
8  }
```

$$1 + \sum_{\texttt{user} \in \texttt{users}} 2 \text{ steps } = 1 + 2 \times |\texttt{users}| \in \theta(|users|)$$

Examples

```
1  public void updateUsers(User[] users)
2  {
3    x = 1;
4    for(user : users)
5    {
6      user.id = x;
7    }
8  }
```

$$1 + \sum_{\texttt{user} \in \texttt{users}} 2 \text{ steps} = 1 + 2 \times |\texttt{users}| \in \theta(N)$$

Examples

```
1  public void userFullName(User[] users, int id)
2  {
3    User user = users[id];
4    String fullName = user.firstName + user.lastName;
5    return fullName;
6  }
```

Examples

```
1  public void userFullName(User[] users, int id)
2  {
3    User user = users[id];
4    String fullName = user.firstName + user.lastName;
5    return fullName;
6  }
```

3

Examples

```
1  public void userFullName(User[] users, int id)
2  {
3    User user = users[id];
4    String fullName = user.firstName + user.lastName;
5    return fullName;
6  }
```

$$3 \in \theta(1)$$

Count the Steps

```
1   public void totalReads(User[] users, Post[] posts)
2   {
3     int totalReads = 0;
4     for(post : posts)
5     {
6       int userReads = 0;
7       for(user : users)
8       {
9         if(user.readPost(post)){ userReads += 1; }
10      }
11      totalReads += userReads;
12    }
13  }
```

$$1 + \sum_{\text{post} \in \text{posts}} \left(3 + \sum_{\text{user} \in \text{users}} 2 \right)$$

Count the Steps

$$1 + \sum_{\texttt{post} \in \texttt{posts}} \left(3 + \sum_{\texttt{user} \in \texttt{users}} 2 \right)$$

Count the Steps

$$1 + \sum_{\texttt{post} \in \texttt{posts}} \left(3 + \sum_{\texttt{user} \in \texttt{users}} 2 \right)$$

$$= 1 + \sum_{\texttt{post} \in \texttt{posts}} (3 + 2 \cdot |\texttt{users}|)$$

Count the Steps

$$1 + \sum_{\texttt{post} \in \texttt{posts}} \left(3 + \sum_{\texttt{user} \in \texttt{users}} 2 \right)$$

$$= 1 + \sum_{\texttt{post} \in \texttt{posts}} (3 + 2 \cdot |\texttt{users}|)$$

$$= 1 + \left(\sum_{\texttt{post} \in \texttt{posts}} 3 \right) + \left(\sum_{\texttt{post} \in \texttt{posts}} 2 \cdot |\texttt{users}| \right)$$

Count the Steps

$$1 + \sum_{\texttt{post} \in \texttt{posts}} \left(3 + \sum_{\texttt{user} \in \texttt{users}} 2 \right)$$

$$= 1 + \sum_{\texttt{post} \in \texttt{posts}} (3 + 2 \cdot |\texttt{users}|)$$

$$= 1 + \left(\sum_{\texttt{post} \in \texttt{posts}} 3 \right) + \left(\sum_{\texttt{post} \in \texttt{posts}} 2 \cdot |\texttt{users}| \right)$$

$$= 1 + (3 \cdot |\texttt{posts}|) + \left(\sum_{\texttt{post} \in \texttt{posts}} 2 \cdot |\texttt{users}| \right)$$

Count the Steps

$$1 + \sum_{\texttt{post} \in \texttt{posts}} \left(3 + \sum_{\texttt{user} \in \texttt{users}} 2 \right)$$

$$= 1 + \sum_{\texttt{post} \in \texttt{posts}} (3 + 2 \cdot |\texttt{users}|)$$

$$= 1 + \left(\sum_{\texttt{post} \in \texttt{posts}} 3 \right) + \left(\sum_{\texttt{post} \in \texttt{posts}} 2 \cdot |\texttt{users}| \right)$$

$$= 1 + (3 \cdot |\texttt{posts}|) + \left(\sum_{\texttt{post} \in \texttt{posts}} 2 \cdot |\texttt{users}| \right)$$

$$= 1 + (3 \cdot |\texttt{posts}|) + (2 \cdot |\texttt{users}| \cdot |\texttt{posts}|)$$

Count the Steps

$$1 + \sum_{\texttt{post} \in \texttt{posts}} \left(3 + \sum_{\texttt{user} \in \texttt{users}} 2 \right)$$

$$= 1 + \sum_{\texttt{post} \in \texttt{posts}} (3 + 2 \cdot |\texttt{users}|)$$

$$= 1 + \left(\sum_{\texttt{post} \in \texttt{posts}} 3 \right) + \left(\sum_{\texttt{post} \in \texttt{posts}} 2 \cdot |\texttt{users}| \right)$$

$$= 1 + (3 \cdot |\texttt{posts}|) + \left(\sum_{\texttt{post} \in \texttt{posts}} 2 \cdot |\texttt{users}| \right)$$

$$= 1 + (3 \cdot |\texttt{posts}|) + (2 \cdot |\texttt{users}| \cdot |\texttt{posts}|)$$

$$\in \theta(|\texttt{users}| \cdot |\texttt{posts}|)$$

Another Example

```
1   public int myAlgorithm(int[] input)
2   {
3     if(input.size % 2 == 0){
4       return 12345;
5     } else {
6       var total = 0;
7       for(i : input)
8       {
9         total += i;
10      }
11      return total;
12    }
13  }
```

Another Example

```
1   public int myAlgorithm(int[] input)
2   {
3     if(input.size % 2 == 0){
4       θ(1)
5     } else {
6       θ(1)
7       for(i : input)
8       {
9         θ(1)
10      }
11      θ(1)
12    }
13  }
```

Another Example

```
1   public int myAlgorithm(int[] input)
2   {
3     if(input.size % 2 == 0){
4       θ(1)
5     } else {
6       θ(1)
7       for(i : input)
8       {
9         θ(1)
10      }
11      θ(1)
12    }
13  }
```

Let's call $|input| = N$

Another Example

```
1   public int myAlgorithm(int[] input)
2   {
3     if(input.size % 2 == 0){
4       θ(1)
5     } else {
6       θ(1)
7       θ(N · 1)
8       θ(1)
9     }
10  }
```

Another Example

```
1  public int myAlgorithm(int[] input)
2  {
3    if(input.size % 2 == 0){
4      θ(1)
5    } else {
6      θ(N)
7    }
8  }
```

Another Example

```
1    public int myAlgorithm(int[] input)
2    {
3      if(input.size % 2 == 0){
4        θ(1)
5      } else {
6        θ(N)
7      }
8    }
```

$\theta(1)$ if N is even **OR** $\theta(N)$ if N is odd.

Multi-Class Functions

$$T(N) = \begin{cases} \theta(1) & \textbf{if } N \text{ is even} \\ \theta(N) & \textbf{if } N \text{ is odd} \end{cases}$$

Multi-Class Functions

$$T(N) = \begin{cases} \theta(1) & \textbf{if } N \text{ is even} \\ \theta(N) & \textbf{if } N \text{ is odd} \end{cases}$$

What is the complexity class of $T(N)$?

Multi-Class Functions

$$T(N) = \begin{cases} \theta(1) & \textbf{if } N \text{ is even} \\ \theta(N) & \textbf{if } N \text{ is odd} \end{cases}$$

What is the complexity class of $T(N)$?

- $T(N) \in O(N)$ is a **tight** bound.
- $T(N) \in \Omega(1)$ is a **tight** bound.

Multi-Class Functions

$$T(N) = \begin{cases} \theta(1) & \textbf{if } N \text{ is even} \\ \theta(N) & \textbf{if } N \text{ is odd} \end{cases}$$

What is the complexity class of $T(N)$?

- $T(N) \in O(N)$ is a **tight** bound.
- $T(N) \in \Omega(1)$ is a **tight** bound.

If the tight Big-O and Big-Ω bounds are different, the function is not in ANY complexity class.

Multi-Class Functions

$$T(N) = \begin{cases} \theta(1) & \textbf{if } N \text{ is even} \\ \theta(N) & \textbf{if } N \text{ is odd} \end{cases}$$

What is the complexity class of $T(N)$?

- $T(N) \in O(N)$ is a **tight** bound.
- $T(N) \in \Omega(1)$ is a **tight** bound.

If the tight Big-O and Big-Ω bounds are different, the function is not in ANY complexity class. (Big-Theta doesn't exist).

Does Big-Theta Exist?

$N + 2N^2$ belongs to one complexity class. $(\theta(N^2))$

Does Big-Theta Exist?

$N + 2N^2$ belongs to one complexity class. $(\theta(N^2))$

$5N + 10N^2 + 2^N$ belongs to one complexity class $(\theta(2^N))$

Does Big-Theta Exist?

$N + 2N^2$ belongs to one complexity class. $(\theta(N^2))$

$5N + 10N^2 + 2^N$ belongs to one complexity class $(\theta(2^N))$

$$\begin{cases} 2^N & \textbf{if } \texttt{rand()} > 0.5 \\ N & \textbf{otherwise} \end{cases}$$ does **not** belong to one complexity class.

Does Big-Theta Exist?

$N + 2N^2$ belongs to one complexity class. $(\theta(N^2))$

$5N + 10N^2 + 2^N$ belongs to one complexity class $(\theta(2^N))$

$$\begin{cases} 2^N & \textbf{if } \texttt{rand()} > 0.5 \\ N & \textbf{otherwise} \end{cases}$$ does **not** belong to one complexity class.

- **Usually $\theta(f_1(N) + f_2(N) + \ldots)$ is based on the dominant term**
- **If you see cases (i.e., '{'), it's probably multi-class.**

In practice...

Most documentation uses Big-O (upper, 'worst-case') bounds.

In practice...

Most documentation uses Big-O (upper, 'worst-case') bounds.

- There's always a Big-O bound.

In practice...

Most documentation uses Big-O (upper, 'worst-case') bounds.

- There's always a Big-O bound.
- The best case usually doesn't bring down production servers.

Bubblesort

4 10 9 18 20 6 3 14 13 7

Bubblesort

Bubblesort

Bubblesort

4	10	9	18	20	6	3	14	7	13

Bubblesort

Bubblesort

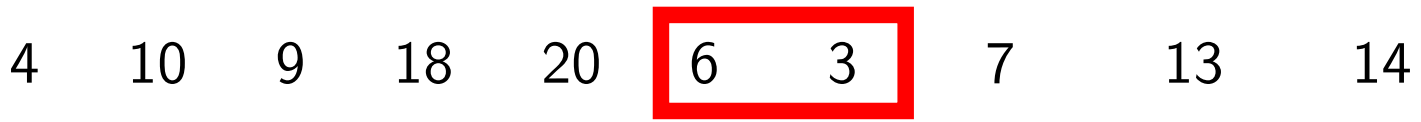
Bubblesort

Bubblesort

Bubblesort

Bubblesort

Bubblesort



Bubblesort

4 10 9 18 20 3 6 7 13 14

Bubblesort

4 10 9 18 20 3 6 7 13 14

Bubblesort

Bubblesort

Bubblesort

```
1   public void bubblesort(int[] data)
2   {
3     int N = data.length;
4     for(int i = N - 2; i >= 0; i--)
5     {
6       for(int j = i; j <= N - 1; j++)
7       {
8         if(data[j+1] < data[j])
9         {
10          swap data[j] and data[j+1]
11        }
12      }
13    }
14  }
```

Bubblesort

```
1            if(data[j+1] < data[j])
2            {
3              swap data[j] and data[j+1]
4            }
```

Bubblesort

```
1            if(data[j+1] < data[j])
2            {
3              swap data[j] and data[j+1]
4            }
```

$\theta(1)$

Bubblesort

```
1   public void bubblesort(int[] data)
2   {
3     int N = data.length;
4     for(int i = N - 2; i >= 0; i--)
5     {
6       for(int j = i; j <= N - 1; j++)
7       {
8         θ(1)
9       }
10    }
11  }
```

Bubblesort

```
1    public void bubblesort(int[] data)
2    {
3      int N = data.length;
4      for(int i = N - 2; i >= 0; i--)
5      {
6        ∑_{j=i}^{N-1} θ(1)
7      }
8    }
```

Wait, what...?

$\theta(1)$ is a set of functions (or complexity class).

$\sum_{j=i}^{N-1} \theta(1) = \sum_{j=i}^{N-1} f(N)$ where we know $f(N) \in \theta(1)$

Bubblesort

```
1    public void bubblesort(int[] data)
2    {
3      int N = data.length;
4      for(int i = N - 2; i >= 0; i--)
5      {
6        ∑_{j=i}^{N-1} θ(1)
7      }
8    }
```

Bubblesort

```
1    public void bubblesort(int[] data)
2    {
3      int N = data.length;
4      for(int i = N - 2; i >= 0; i--)
5      {
6        (N − 1 + 1 − i) · θ(1)
7      }
8    }
```

Bubblesort

```
1    public void bubblesort(int[] data)
2    {
3      int N = data.length;
4      for(int i = N - 2; i >= 0; i--)
5      {
6        (N − i) · θ(1)
7      }
8    }
```

Bubblesort

```
1    public void bubblesort(int[] data)
2    {
3      int N = data.length;
4      ∑_{i=0}^{N-2} (N − i) · θ(1)
5    }
```

Bubblesort

```
1    public void bubblesort(int[] data)
2    {
3      int N = data.length;
4      $\left(\sum_{i=0}^{N-2} N \cdot \theta(1)\right) - \left(\sum_{i=0}^{N-2} i \cdot \theta(1)\right)$
5    }
```

Bubblesort

```
1    public void bubblesort(int[] data)
2    {
3      int N = data.length;
4      ((N − 2 + 1 − 0) · N · θ(1)) − (∑_{i=0}^{N−2} i · θ(1))
5    }
```

Bubblesort

```
1    public void bubblesort(int[] data)
2    {
3      int N = data.length;
4      (N^2 − N) · θ(1) − (∑_{i=0}^{N−2} i · θ(1))
5    }
```

Bubblesort

```
1    public void bubblesort(int[] data)
2    {
3      int N = data.length;
4      (N^2 − N) · θ(1) − (0 + ((N−2)(N−2+1))/2 · θ(1))
5    }
```

Bubblesort

```
1    public void bubblesort(int[] data)
2    {
3      int N = data.length;
4      (N^2 − N) · θ(1) − ( (N^2 − 3N + 2)/2 · θ(1) )
5    }
```

Bubblesort

```
1    public void bubblesort(int[] data)
2    {
3      int N = data.length;
4      (1/2 N^2 + N - 2) · θ(1)
5    }
```

Algebra with θ

Let $f'(N) = c \cdot f(N)$ where $f(N) \in \theta(g(N))$
(c is a constant)

Algebra with θ

Let $f'(N) = c \cdot f(N)$ where $f(N) \in \theta(g(N))$
(c is a constant)

What complexity class is $f'(N)$ in?

Algebra with θ

Let $f'(N) = c \cdot f(N)$ where $f(N) \in \theta(g(N))$
(c is a constant)

What complexity class is $f'(N)$ in?

So $c \cdot \theta(g(N)) = \theta(g(N))$

Algebra with θ

$$c \cdot \theta(f(N)) \quad =$$

Algebra with θ

$$c \cdot \theta(f(N)) \quad = \quad \theta(f(N))$$

Algebra with θ

$$c \cdot \theta(f(N)) = \theta(f(N))$$

$$N \cdot \theta(f(N)) =$$

Algebra with θ

$$c \cdot \theta(f(N)) = \theta(f(N))$$

$$N \cdot \theta(f(N)) = \theta(N \cdot f(N))$$

Algebra with θ

$$c \cdot \theta(f(N)) = \theta(f(N))$$

$$N \cdot \theta(f(N)) = \theta(N \cdot f(N))$$

$$g(N) \cdot \theta(f(N)) =$$

Algebra with θ

$$
\begin{aligned}
c \cdot \theta(f(N)) &= \theta(f(N)) \\
N \cdot \theta(f(N)) &= \theta(N \cdot f(N)) \\
g(N) \cdot \theta(f(N)) &= \theta(g(N) \cdot f(N)) \quad (\text{if } \theta(g(N)) \text{ exists})
\end{aligned}
$$

Algebra with θ

$$
\begin{aligned}
c \cdot \theta(f(N)) &= \theta(f(N)) \\
N \cdot \theta(f(N)) &= \theta(N \cdot f(N)) \\
g(N) \cdot \theta(f(N)) &= \theta(g(N) \cdot f(N)) \quad (\text{if } \theta(g(N)) \text{ exists}) \\
\theta(g(N)) + \theta(f(N)) &=
\end{aligned}
$$

Algebra with θ

$$
\begin{aligned}
c \cdot \theta(f(N)) &= \theta(f(N)) \\
N \cdot \theta(f(N)) &= \theta(N \cdot f(N)) \\
g(N) \cdot \theta(f(N)) &= \theta(g(N) \cdot f(N)) \quad \text{(if } \theta(g(N)) \text{ exists)} \\
\theta(g(N)) + \theta(f(N)) &= \theta(g(N) + f(N))
\end{aligned}
$$

Algebra with θ

$$
\begin{aligned}
c \cdot \theta(f(N)) &= \theta(f(N)) \\
N \cdot \theta(f(N)) &= \theta(N \cdot f(N)) \\
g(N) \cdot \theta(f(N)) &= \theta(g(N) \cdot f(N)) \quad \text{(if } \theta(g(N)) \text{ exists)} \\
\theta(g(N)) + \theta(f(N)) &= \theta(g(N) + f(N)) \\
&= \text{The greater of } \theta(f(N)) \textbf{ or } \theta(g(N))
\end{aligned}
$$

Bubblesort

```
1    public void bubblesort(int[] data)
2    {
3      int N = data.length;
4      (1/2 N^2 + N - 2) · θ(1)
5    }
```

Bubblesort

```
1    public void bubblesort(int[] data)
2    {
3      int N = data.length;
4      θ((1/2 N^2 + N − 2) · 1)
5    }
```

Bubblesort

```
1    public void bubblesort(int[] data)
2    {
3      int N = data.length;
4      θ(½N² + N − 2)
5    }
```

Bubblesort

```
1    public void bubblesort(int[] data)
2    {
3      int N = data.length;
4      θ(½N²)
5    }
```

Bubblesort

```
1    public void bubblesort(int[] data)
2    {
3      int N = data.length;
4      θ(N^2)
5    }
```

Bubblesort

```
1    public void bubblesort(int[] data)
2    {
3      θ(1)
4      θ(N^2)
5    }
```

Bubblesort

```
1    public void bubblesort(int[] data)
2    {
3      θ(N^2)
4    }
```

Bubblesort

```
1    public void bubblesort(int[] data)
2    {
3      θ(N^2)
4    }
```

Bubblesort on an array is $\theta(N^2)$

Rules of Thumb

- **Lines of Code**: Add Complexities
- **Loops**: Multiply Complexities
- **If/Then**: Cases block '{'

Bubblesort

```
1   public void bubblesort(List[Integer] data)
2   {
3     int N = data.size();
4     for(int i = N - 2; i >= 0; i--)
5     {
6       for(int j = i; j <= N - 1; j++)
7       {
8         if(data.get(j+1) < data.get(j))
9         {
10          int temp = data.get(j);
11          data.set(j, data.get(j+1));
12          data.set(j+1, temp);
13        }
14      }
15    }
16  }
```

Lists

A java `List` can be a:

- **Linked List**: `LinkedList`
  - `data.get(x)`, `data.set(x)` are
- **Vector Array**: `ArrayList`
  - `data.get(x)`, `data.set(x)` are

Lists

A java `List` can be a:

- **Linked List**: `LinkedList`
  - `data.get(x)`, `data.set(x)` are $O(N)$
- **Vector Array**: `ArrayList`
  - `data.get(x)`, `data.set(x)` are

Lists

A java `List` can be a:

- **Linked List**: `LinkedList`
  - `data.get(x)`, `data.set(x)` are $O(N)$[1]
- **Vector Array**: `ArrayList`
  - `data.get(x)`, `data.set(x)` are

[1] $\theta(x)$ would be more precise, but there's no better bound in terms of N.

Lists

A java `List` can be a:

- **Linked List**: `LinkedList`
  - `data.get(x)`, `data.set(x)` are $O(N)$[1]
- **Vector Array**: `ArrayList`
  - `data.get(x)`, `data.set(x)` are $\theta(1)$

[1] $\theta(x)$ would be more precise, but there's no better bound in terms of N.

Lists

A java `List` can be a:

- **Linked List**: `LinkedList`
  - `data.get(x)`, `data.set(x)` are $O(N)$[1]
- **Vector Array**: `ArrayList`
  - `data.get(x)`, `data.set(x)` are $\theta(1)$ (this implies $O(1)$)

[1] $\theta(x)$ would be more precise, but there's no better bound in terms of N.

Lists

A java `List` can be a:

- **Linked List**: `LinkedList`
  - `data.get(x)`, `data.set(x)` are $O(N)$[1]
- **Vector Array**: `ArrayList`
  - `data.get(x)`, `data.set(x)` are $\theta(1)$ (this implies $O(1)$)

$$T_{\texttt{get}}(N) = T_{\texttt{set}}(N) = \begin{cases} O(1) & \textbf{if } \texttt{data} \text{ is a } \texttt{ArrayList} \\ O(N) & \textbf{if } \texttt{data} \text{ is a } \texttt{LinkedList} \end{cases}$$

[1] $\theta(x)$ would be more precise, but there's no better bound in terms of N.

Lists

A java `List` can be a:

- **Linked List**: `LinkedList`
  - `data.get(x)`, `data.set(x)` are $O(N)$[1]
- **Vector Array**: `ArrayList`
  - `data.get(x)`, `data.set(x)` are $\theta(1)$ (this implies $O(1)$)

$$T_{\texttt{get}}(N) = T_{\texttt{set}}(N) = \begin{cases} O(1) & \textbf{if } \texttt{data} \text{ is a } \texttt{ArrayList} \\ O(N) & \textbf{if } \texttt{data} \text{ is a } \texttt{LinkedList} \end{cases}$$

$$T_{\texttt{get}}(N) = T_{\texttt{set}}(N) = ???$$

[1] $\theta(x)$ would be more precise, but there's no better bound in terms of N.

Lists

A java `List` can be a:

- **Linked List**: `LinkedList`
  - `data.get(x)`, `data.set(x)` are $O(N)$[1]
- **Vector Array**: `ArrayList`
  - `data.get(x)`, `data.set(x)` are $\theta(1)$ (this implies $O(1)$)

$$T_{\texttt{get}}(N) = T_{\texttt{set}}(N) = \begin{cases} O(1) & \textbf{if } \texttt{data} \text{ is a } \texttt{ArrayList} \\ O(N) & \textbf{if } \texttt{data} \text{ is a } \texttt{LinkedList} \end{cases}$$

$$T_{\texttt{get}}(N) = T_{\texttt{set}}(N) = O(N)$$

[1] $\theta(x)$ would be more precise, but there's no better bound in terms of N.

Algebra with O

$$c \cdot O(f(N)) \quad =$$

Algebra with O

$$c \cdot O(f(N)) \quad = \quad O(f(N))$$

Algebra with O

$$c \cdot O(f(N)) = O(f(N))$$

$$N \cdot O(f(N)) =$$

Algebra with O

$$c \cdot O(f(N)) = O(f(N))$$

$$N \cdot O(f(N)) = O(N \cdot f(N))$$

Algebra with O

$$c \cdot O(f(N)) = O(f(N))$$

$$N \cdot O(f(N)) = O(N \cdot f(N))$$

$$g(N) \cdot O(f(N)) =$$

Algebra with O

$$c \cdot O(f(N)) = O(f(N))$$

$$N \cdot O(f(N)) = O(N \cdot f(N))$$

$$g(N) \cdot O(f(N)) = O(g(N) \cdot f(N))$$

Algebra with O

$$c \cdot O(f(N)) = O(f(N))$$

$$N \cdot O(f(N)) = O(N \cdot f(N))$$

$$g(N) \cdot O(f(N)) = O(g(N) \cdot f(N))$$

$$O(g(N)) + O(f(N)) =$$

Algebra with O

$$c \cdot O(f(N)) = O(f(N))$$

$$N \cdot O(f(N)) = O(N \cdot f(N))$$

$$g(N) \cdot O(f(N)) = O(g(N) \cdot f(N))$$

$$O(g(N)) + O(f(N)) = O(g(N) + f(N))$$

Algebra with O

$$c \cdot O(f(N)) = O(f(N))$$

$$N \cdot O(f(N)) = O(N \cdot f(N))$$

$$g(N) \cdot O(f(N)) = O(g(N) \cdot f(N))$$

$$O(g(N)) + O(f(N)) = O(g(N) + f(N))$$

$$= \text{the greater of } O(f(N)) \textbf{ or } O(g(N))$$

$$\begin{cases} O(g(N)) & \textbf{if one thing} \\ O(f(N)) & \textbf{otherwise} \end{cases} =$$

Algebra with O

$$c \cdot O(f(N)) = O(f(N))$$

$$N \cdot O(f(N)) = O(N \cdot f(N))$$

$$g(N) \cdot O(f(N)) = O(g(N) \cdot f(N))$$

$$O(g(N)) + O(f(N)) = O(g(N) + f(N))$$
$$= \text{the greater of } O(f(N)) \textbf{ or } O(g(N))$$

$$\begin{cases} O(g(N)) & \textbf{if one thing} \\ O(f(N)) & \textbf{otherwise} \end{cases} = \text{the greater of } O(f(N)) \textbf{ or } O(g(N))$$

Algebra with O

$$
\begin{aligned}
c \cdot O(f(N)) &= O(f(N)) \\
N \cdot O(f(N)) &= O(N \cdot f(N)) \\
g(N) \cdot O(f(N)) &= O(g(N) \cdot f(N)) \\
O(g(N)) + O(f(N)) &= O(g(N) + f(N)) \\
&= \text{the greater of } O(f(N)) \textbf{ or } O(g(N)) \\
\begin{cases} O(g(N)) & \textbf{if one thing} \\ O(f(N)) & \textbf{otherwise} \end{cases} &= \text{the greater of } O(f(N)) \textbf{ or } O(g(N)) \\
&= O(g(N) + f(N))
\end{aligned}
$$

Algebra with Ω

$$c \cdot \Omega(f(N)) = \Omega(f(N))$$

$$N \cdot \Omega(f(N)) = \Omega(N \cdot f(N))$$

$$g(N) \cdot \Omega(f(N)) = \Omega(g(N) \cdot f(N))$$

$$\Omega(g(N)) + \Omega(f(N)) = \Omega(g(N) + f(N))$$
$$= \text{the greater of } \Omega(f(N)) \textbf{ or } \Omega(g(N))$$

$$\begin{cases} \Omega(g(N)) & \textbf{if one thing} \\ \Omega(f(N)) & \textbf{otherwise} \end{cases} = \text{the lesser of } \Omega(f(N)) \textbf{ or } \Omega(g(N))$$

Bubblesort on Lists

```
1   public void bubblesort(List[Integer] data)
2   {
3     int N = data.size();
4     for(int i = N - 2; i >= 0; i--)
5     {
6       for(int j = i; j <= N - 1; j++)
7       {
8         if(data.get(j+1) < data.get(j))
9         {
10          int temp = data.get(j);
11          data.set(j, data.get(j+1));
12          data.set(j+1, temp);
13        }
14      }
15    }
16  }
```

Bubblesort on Lists

```
1   public void bubblesort(List[Integer] data)
2   {
3     int N = data.size();
4     for(int i = N - 2; i >= 0; i--)
5     {
6       for(int j = i; j <= N - 1; j++)
7       {
8         if(data.get(j+1) < data.get(j))
9         {
10          O(N)
11          O(N + N)
12          O(N)
13        }
14      }
15    }
16  }
```

Bubblesort on Lists

```
1   public void bubblesort(List[Integer] data)
2   {
3     int N = data.size();
4     for(int i = N - 2; i >= 0; i--)
5     {
6       for(int j = i; j <= N - 1; j++)
7       {
8         if(data.get(j+1) < data.get(j))
9         {
10          O(N)
11        }
12      }
13    }
14  }
```

Bubblesort on Lists

```
1   public void bubblesort(List[Integer] data)
2   {
3     int N = data.size();
4     for(int i = N - 2; i >= 0; i--)
5     {
6       for(int j = i; j <= N - 1; j++)
7       {
8         if(data.get(j+1) < data.get(j))
9         {
10          O(N)
11        } else {
12          O(1)
13        }
14      }
15    }
16  }
```

Bubblesort on Lists

```
1   public void bubblesort(List[Integer] data)
2   {
3     int N = data.size();
4     for(int i = N - 2; i >= 0; i--)
5     {
6       for(int j = i; j <= N - 1; j++)
7       {
8         O(2N) + O(N OR 1)
9       }
10    }
11  }
```

Bubblesort on Lists

```
1   public void bubblesort(List[Integer] data)
2   {
3     int N = data.size();
4     for(int i = N - 2; i >= 0; i--)
5     {
6       for(int j = i; j <= N - 1; j++)
7       {
8         O(2N) + O(N + 1)
9       }
10    }
11  }
```

Bubblesort on Lists

```
1     public void bubblesort(List[Integer] data)
2     {
3       int N = data.size();
4       for(int i = N - 2; i >= 0; i--)
5       {
6         for(int j = i; j <= N - 1; j++)
7         {
8           O(2N) + O(N)
9         }
10      }
11    }
```

Bubblesort on Lists

```
1   public void bubblesort(List[Integer] data)
2   {
3     int N = data.size();
4     for(int i = N - 2; i >= 0; i--)
5     {
6       for(int j = i; j <= N - 1; j++)
7       {
8         O(N)
9       }
10    }
11  }
```

Bubblesort on Lists

```
1   public void bubblesort(List[Integer] data)
2   {
3     int N = data.size();
4     for(int i = N - 2; i >= 0; i--)
5     {
6       ∑_{j=i}^{N-1} O(N)
7     }
8   }
```

Bubblesort on Lists

```
1   public void bubblesort(List[Integer] data)
2   {
3     int N = data.size();
4     for(int i = N - 2; i >= 0; i--)
5     {
6       (N − 1 + 1 − i)O(N)
7     }
8   }
```

Bubblesort on Lists

```
1    public void bubblesort(List[Integer] data)
2    {
3      int N = data.size();
4      for(int i = N - 2; i >= 0; i--)
5      {
6        (N − i)O(N)
7      }
8    }
```

Bubblesort on Lists

```
1    public void bubblesort(List[Integer] data)
2    {
3      int N = data.size();
4      ∑_{i=0}^{N-2} (N − i) · O(N)
5    }
```

Bubblesort on Lists

```
1    public void bubblesort(List[Integer] data)
2    {
3      int N = data.size();
4      (∑_{i=0}^{N-2} N · O(N)) − (∑_{i=0}^{N-2} i · O(N))
5    }
```

Bubblesort on Lists

```
1    public void bubblesort(List[Integer] data)
2    {
3      int N = data.size();
4      ((N − 2 + 1 − 0) · N · O(N)) − (∑_{i=0}^{N−2} i · O(N))
5    }
```

Bubblesort on Lists

```
1    public void bubblesort(List[Integer] data)
2    {
3      int N = data.size();
4      (N^2 − N) · O(N) − O(N) · (∑_{i=0}^{N−2} i)
5    }
```

Bubblesort on Lists

```
1    public void bubblesort(List[Integer] data)
2    {
3      int N = data.size();
4      (N^2 − N) · O(N) − O(N) · ( ((N−2)(N−2+1)) / 2 )
5    }
```

Bubblesort on Lists

```
1   public void bubblesort(List[Integer] data)
2   {
3     int N = data.size();
4     (N^2 − N) · O(N) − O(N) · ( (N^2 − 3N + 2) / 2 )
5   }
```

Bubblesort on Lists

```
1    public void bubblesort(List[Integer] data)
2    {
3      int N = data.size();
4      (½N² + ½N − 1) · O(N)
5    }
```

Bubblesort on Lists

```
1    public void bubblesort(List[Integer] data)
2    {
3      int N = data.size();
4      O(N^3)
5    }
```

Bubblesort on Lists

```
1    public void bubblesort(List[Integer] data)
2    {
3       O(1)
4       O(N^3)
5    }
```

Bubblesort on Lists

```
1    public void bubblesort(List[Integer] data)
2    {
3      O(N^3)
4    }
```

Bubblesort on Lists

Can we do better?

Bubblesort on Lists

```
1     public void bubblesort(List[Integer] data)
2     {
3       int[] array = data.toArray()
4       bubblesort(array) // Use the array implementation
5       data.clear()
6       data.addAll(Arrays.toList(array))
7     }
```

Bubblesort on Lists

```
1     public void bubblesort(List[Integer] data)
2     {
3       O(N)
4       bubblesort(array) // Use the array implementation
5       data.clear()
6       data.addAll(Arrays.toList(array))
7     }
```

Bubblesort on Lists

```
1   public void bubblesort(List[Integer] data)
2   {
3     O(N)
4     O(N^2)
5     data.clear()
6     data.addAll(Arrays.toList(array))
7   }
```

Bubblesort on Lists

```
1   public void bubblesort(List[Integer] data)
2   {
3     O(N)
4     O(N^2)
5     O(N)
6     data.addAll(Arrays.toList(array))
7   }
```

Bubblesort on Lists

```
1   public void bubblesort(List[Integer] data)
2   {
3     O(N)
4     O(N^2)
5     O(N)
6     O(N)
7   }
```

Bubblesort on Lists

```
1    public void bubblesort(List[Integer] data)
2    {
3       O(N + N^2 + N + N)
4    }
```

Bubblesort on Lists

```
1    public void bubblesort(List[Integer] data)
2    {
3      O(N^2)
4    }
```

Bubblesort on Lists

```
1    public void bubblesort(List[Integer] data)
2    {
3       O(N^2)
4    }
```

Organizing data first can make code faster