Compsci 201
Linked Lists, Big-Oh, Markov
(and interview questions)

Susan Rodger
February 21, 2020

Yes it did snow!

L is for …

- Loops
  - Iteration is a wonderful thing
- Library
  - Where we find APIs rather than books
- Linked Lists
  - From Node to Node

Announcements

- Exam 1 – Do not discuss until with anyone until handed back
- APT Quiz 1 must complete by Monday
  - Do by yourself
- Assignment P3 out today – due 2/27
  - Builds on P2 Markov
- Discussion 2/24
  - P3 and Linked Lists APTS
First Quick Review of Linked Lists

- Interview Questions
  - Big-Oh, APT practice, APT Practice

- Linked List Review
  - Visualize, Metaphors, Code

- Efficient WordGram
  - Maps and text generation

Visualizing/Understanding Nodes

- [https://coursework.cs.duke.edu/rodger/diyad-new](https://coursework.cs.duke.edu/rodger/diyad-new)
  - `diyad.linkedlist.SimpleLinkedList`
  - Like pair, note: `this` not needed below
  - Instance variables for String and "next node"

```java
private class Node{
    String info;
    Node next;
    public Node(String key, Node link){
        this.info = key;
        this.next = link;
    }
}
```

Iterators to the Rescue

- Iterators are soooo nice. But timing?
  - Why $O(N)$ linked list and $O(N^2)$ array?

```java
public List<String> removeAllIterator(String target, List<String> list) {
    Iterator<String> iter = list.iterator();
    while (iter.hasNext()) {
        String w = iter.next();
        if (w.equals(target)) {
            iter.remove();
        }
    }
    return list;
}
```
From Iterator to Iterable

• Enhanced for: `for(String s : list) { ...`
  • Underneath, uses iterator
  • Code below $O(N)$ for both lists!

```java
public int iterateEach(List<String> list) {
    int total = 0;
    for (String s : list) {
        total += s.length();
    }
    return total;
}
```

2/21/2020 CompSci 201, Spring 2020

Compare the two

• ListSplicer.java (linked list first, then arrayList)

<table>
<thead>
<tr>
<th>iterateEach</th>
<th>iterate</th>
</tr>
</thead>
<tbody>
<tr>
<td>100000</td>
<td>0.0120</td>
</tr>
<tr>
<td>200000</td>
<td>0.0056</td>
</tr>
<tr>
<td>300000</td>
<td>0.0035</td>
</tr>
<tr>
<td>400000</td>
<td>0.0046</td>
</tr>
<tr>
<td>500000</td>
<td>0.0081</td>
</tr>
<tr>
<td>600000</td>
<td>0.0091</td>
</tr>
<tr>
<td>700000</td>
<td>0.0113</td>
</tr>
<tr>
<td>800000</td>
<td>0.0129</td>
</tr>
<tr>
<td>900000</td>
<td>0.0165</td>
</tr>
<tr>
<td>1000000</td>
<td>0.0193</td>
</tr>
</tbody>
</table>

WOTO (Correctness counts)

If you submitted this WOTO last time your entry was deleted!


Linked list too slow with .get
Interview Interlude (à la 201)

- [https://leetcode.com/problems/two-sum/](https://leetcode.com/problems/two-sum/)
- Given an array of integers, return indices \( (j,k) \) of two numbers that add to a target value. There will be one solution, can’t use same element twice.
- Example: `findTwo([2,7,11,15], 9)`
- Returns \([0,1]\)

- Think, pair, share … first idea, quantify O-notation

Big-Oh Analysis

- Do we have to look at every number?
  - Yes, otherwise we might miss the one!

- For \( X \), do we know \( Y \) such that \( X+Y = \text{target} \)?
  - Can we find \( Y \)? Where is it?

- Given \( X \), if we look at all values to find \( Y \) then …
  - How do we search for a value

Goal of an Interview/Interviewer


Running time?
Goal of an Interview/Interviewer


```
public int[] methodA(int[] nums, int target) {
    for (int j = 0; j < nums.length; j++) {
        for (int k = j + 1; k < nums.length; k++) {
            if (nums[j] + nums[k] == target) {
                return new int[] {j, k};
            }
        }
    }
    return new int[] {0, 0}; // never reached
}
```

Running time?

Goal of an Interview/Interviewer


```
public int[] methodA(int[] nums, int target) {
    for (int j = 0; j < nums.length; j++) {
        for (int k = j + 1; k < nums.length; k++) {
            if (nums[j] + nums[k] == target) {
                return new int[] {j, k};
            }
        }
    }
    return new int[] {0, 0}; // never reached
}
```

Running time?

```
O(n^2)
```

Running times in seconds

Machine: 10^9 instructions/sec

<table>
<thead>
<tr>
<th>N</th>
<th>O(log N)</th>
<th>O(N)</th>
<th>O(N log N)</th>
<th>O(N^2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>3E-9</td>
<td>1E-8</td>
<td>3.3E-8</td>
<td>0.0000001</td>
</tr>
<tr>
<td>100</td>
<td>7E-9</td>
<td>1E-7</td>
<td>6.64E-7</td>
<td>0.0001</td>
</tr>
<tr>
<td>1,000</td>
<td>1E-8</td>
<td>1E-6</td>
<td>0.00001</td>
<td>0.001</td>
</tr>
<tr>
<td>10,000</td>
<td>1.3E-8</td>
<td>0.00001</td>
<td>0.0001329</td>
<td>0.102</td>
</tr>
<tr>
<td>100,000</td>
<td>1.7E-8</td>
<td>0.0001</td>
<td>0.001661</td>
<td>10.008</td>
</tr>
<tr>
<td>1,000,000</td>
<td>0.00000002</td>
<td>0.001</td>
<td>0.0199</td>
<td>16.7 min</td>
</tr>
<tr>
<td>1,000,000,000</td>
<td>0.00000003</td>
<td>1.002</td>
<td>65.8</td>
<td>31.8 years</td>
</tr>
</tbody>
</table>
Does efficiency matter?

• Why do we need a copy for binarySearch?

28-29?

26?

24?

Does efficiency matter?

• Why do we need a copy for binarySearch?

28-29?

26?

24?

Does efficiency matter?

• Why do we need a copy for binarySearch?

28-29?

26?

24?

Does efficiency matter?

• Why do we need a copy for binarySearch?

28-29?

26?

24?

• You don’t need to know Java like this

O(n) once each
Does efficiency matter?

- Why do we need a copy for binarySearch?
  - You don't need to know Java like this

28-29?
O(n)
once
each

26?
O(log n)

24?

2/21/2020 CompSci 201, Spring 2020
Does efficiency matter?

• Why do we need a copy for binarySearch?
  • You don’t need to know Java like this

28-29?
O(n)
once
each

26?
O(log n)

24?
O(n log n)

Method B – just larger

```java
public int[] methodB(int[] nums, int target) {
    int[] copy = new int[nums.length];
    ArrayList<Integer> list = new ArrayList<>();
    for (int val: nums) list.add(val);
    System.arraycopy(nums, 0, copy, 0, nums.length);
    Arrays.sort(copy);
    for (int a : copy) {
        int index = Arrays.binarySearch(copy, target - a);
        if (index >= 0) {
            int j = list.indexOf(a);
            int k = list.indexOf(target - a);
            return new int[] {j, k};
        }
    }
    return new int[] {0, 0}; // never reached
}
```

Can we do better?

• Can we search faster?
  50-55:
  59:
  60:
  61:

• Total?
Can we do better?

- Can we search faster than $O(\log N)$?

  ```java
  public int[] methodD(int[] nums, int target) {
      HashSet<Integer> set = new HashSet<>();
      ArrayList<Integer> list = new ArrayList<>();
      for (int val: nums) {
          list.add(val);
          set.add(val);
      }
      for (int a: nums) {
          int other = target - a;
          if (set.contains(other)) {
              int j = list.indexOf(a);
              int k = list.indexOf(other);
              return new int[] {j, k};
          }
      }
      return new int[] {0, 0}; // never reached
  }
  ```

- Total?
Can we do better?

- Can we search faster than $O(\log N)$
  
  50-55: $O(N)$
  
  59: $O(1)$
  
  60:
  
  61:

- Total?

```java
public int [] methodD(int[] nums, int target) {
    HashSet<Integer> set = new HashSet<>();
    ArrayList<Integer> list = new ArrayList<>();
    for (int val: nums) {
        list.add(val);
        set.add(val);
    }
    for (int a: nums) {
        int other = target - a;
        if (set.contains(other)) {
            int j = list.indexOf(a);
            int k = list.indexOf(other);
            return new int [] {j,k};
        }
    }
    return new int [] {0,0}; // never reached
}
```

2/21/2020

Can we do better?

- Can we search faster than $O(\log N)$
  
  50-55: $O(N)$
  
  59: $O(1)$
  
  60:
  
  61:

- Total?

```java
public int [] methodD(int[] nums, int target) {
    HashSet<Integer> set = new HashSet<>();
    ArrayList<Integer> list = new ArrayList<>();
    for (int val: nums) {
        list.add(val);
        set.add(val);
    }
    for (int a: nums) {
        int other = target - a;
        if (set.contains(other)) {
            int j = list.indexOf(a);
            int k = list.indexOf(other);
            return new int [] {j,k};
        }
    }
    return new int [] {0,0}; // never reached
}
```

2/21/2020
Can we do better?

- Can we search faster than $O(\log N)$
  - 50-55: $O(N)$
  - 59: $O(1)$
  - 60: $O(N)$
  - 61: $O(N)$

- Total?

```java
public int[] methodD(int[] nums, int target) {
    Hashset<Integer> set = new HashSet<>();
    ArrayList<Integer> list = new ArrayList<>();
    for (int val: nums) {
        list.add(val);
        set.add(val);
    }
    for (int a: nums) {
        int other = target-a;
        if (set.contains(other)) {
            int j = list.indexOf(a);
            int k = list.indexOf(other);
            return new int[] {j, k};
        }
    }
    return new int[] {0,0}; // never reached
}
```

MethodD larger

```java
public int[] methodD(int[] nums, int target) {
    Hashset<Integer> set = new HashSet<>();
    ArrayList<Integer> list = new ArrayList<>();
    for (int val: nums) {
        list.add(val);
        set.add(val);
    }
    for (int a: nums) {
        int other = target-a;
        if (set.contains(other)) {
            int j = list.indexOf(a);
            int k = list.indexOf(other);
            return new int[] {j, k};
        }
    }
    return new int[] {0,0}; // never reached
}
```
Method C – not discussed

```java
36 @
37 38
39 40
41 42
43 44
45 46
47

public int[] methodC(int[] nums, int target) {
    ArrayList<Integer> list = new ArrayList<>();
    for (int val: nums) list.add(val);
    for (int j=0; j < nums.length; j++) {
        int index = list.indexOf(target - nums[j]);
        if (index != -1) {
            return new int[] {j, index};
        }
    }
    return new int[] {0,0}; // never reached
}
```

Running times of 4 methods on list of size 300000

Each found the two values shown in the array in time listed

- Method A: [84761, 203040] in 6.60907
- Method B: [84761, 203040] in 0.15114
- Method C: [84761, 203040] in 32.38338
- Method D: [84761, 203040] in 0.05609

WOTO


Krysta Svore

- Manages Microsoft Quantum Architectures and Computation Group (QuArC)
- Princeton Math major, CompSci/French minor

“We think a quantum computer could possibly solve these [hard] types of problems in a time frame that’s more reasonable than the life of the universe, maybe a couple of years, or a couple of days, or a couple of seconds,” Svore said. “Exponentially faster.”
Markov 2: Efficiency

- Idea related to machine learning
  - Given a training text, use it to create a model
  - Using the model, generate random text
- Infinite Monkey Theorem?
  - Don't type at random
  - Use letter frequencies!!

Naïve, Brute Force Idea

- Given training text "the theatre through that helps"
  - Generate random text based on frequencies
- For a model-2 Markov process: start with "th"
  - Characters after "th": \{"e", "e", "r", "a"\}
  - Choose one at random, say "e": generate!
    - Now use with "he", since "th" + "e" = "he"
    - Following "he":
  - Why naïve? Re-scan text every time for follows
Naïve, Brute Force Idea

- Given training text "the theatre through that helps"
  - Generate random text based on frequencies
- For a model-2 Markov process: start with "th"
  - Characters after "th": {"e", "e", "r", "a"}
  - Choose one at random, say "e": generate!
    - Now use with "he", since "th" + "e" = "he"
    - Following "he": {"", "a", "l"}
- Why naïve? Re-scan text every time for follows

Finding Follow Characters

- Scan entire text looking for key
  - [Link]
- Loop O(T) for myText with T characters
  - Again?

Conceptual and Analytical O(T)

- To find every follow character for "th" or N-gram
  - Scan text looking for "th", when found ...?
  - Repeat, but start scanning from after "th" found
  - In code, scanning means call .indexOf ..
    - But with a parameter of where to start search
- Does this look at all T characters?
  - More than once?

Don't Scan N times, Scan Once

- We generate N random characters
  - Get follows N times, each O(T), total is O(NT)
- Suppose we find all N-grams, e.g., 2-grams
  - "th" -> {"e", "e", "r", "a"}
  - "he" -> {"", "a", "l"}
  - ...
- Map of 2-gram to ArrayList of following chars
  - Create in O(T) time. Get follows is O(1)
    - So total is O(N + T)
Inheritance

• In BaseMarkov two methods
  • `generateRandomText` calls `getFollows`

• EfficientMarkov extends BaseMarkov
  • Inherits all of `BaseMarkov` methods
  • Re-implements or overrides `getFollows`
  • Inherited `generatedRandomText`
    • calls new `getFollows`, overridden method!!

Efficient Markov

• Started it for you …

```java
import java.util.*;

public class EfficientMarkov extends BaseMarkov {
    private Map<String, ArrayList<String>> myMap;
}
```

Markov Big Picture

• Use `BaseMarkov` as a start, create `EfficientMarkov`
  • Make constructors work, create map
  • @Override `getFollows` to be O(1) not O(T)
  • Benchmark these programs

• Use `WordGram` rather than `String`
  • Generate word-based random text, not char
  • String is collection of characters, `WordGram` is collection of Strings
  • Use same idea for map, but use `WordGram`

WOTO