EE422C – Assignment 3

Poetic Walks

Problem Statement:

Graphs — what are they good for? Poetry!

In this lab, we will explore a common Natural Language Processing technique known as N-gram.

Write a program that completes the poetry with bridge words.

For example, here's a small but inspirational corpus:

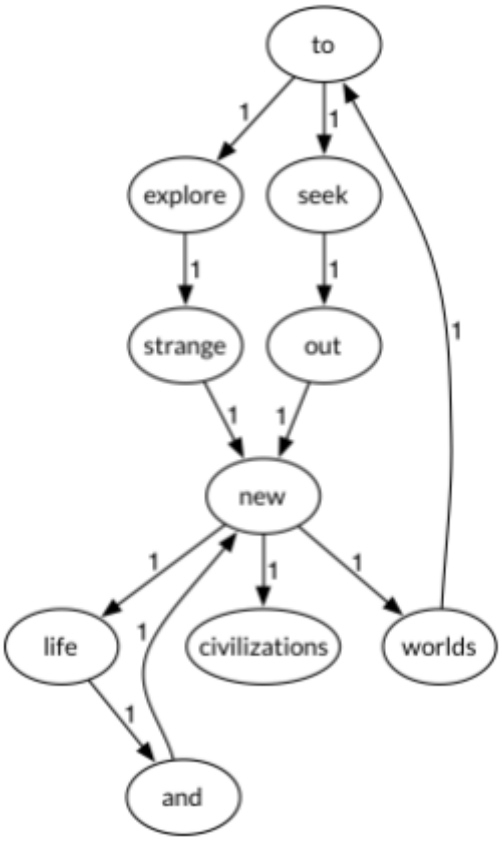
Poem Input:

To explore strange new worlds
To seek out new life and new civilizations

`GraphPoet` will use this corpus to generate a word affinity graph where:

- vertices are case-insensitive words, and
- edge weights are in-order adjacency counts.

The graph is shown below. In this example, all the edges have weight 1 because no word pair is repeated.



From there, given this dry bit of business-speak input:

Input:

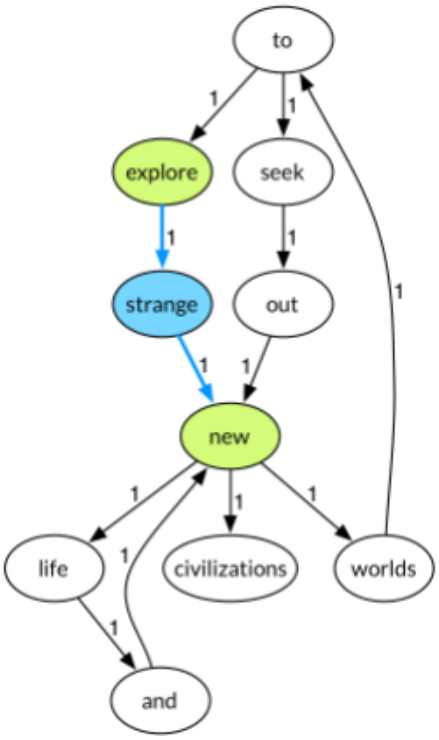
→ *Seek to explore new and exciting synergies!*

The poet will use the graph to generate a poem by inserting (where possible) a *bridge word* between each pair of input words given the poem input:

The resulting output:

> Seek to explore **strange** new **life** and exciting synergies!

w_1		w_2	path?	bridge word
seek	→	*to*	no two-edge-long path from *seek* to *to*	—
to	→	*explore*	no two-edge-long path from *to* to *explore*	—
explore	→	*new*	one two-edge-long path from *explore* to *new* with weight 2	*strange*
new	→	*and*	one two-edge-long path from *new* to *and* with weight 2	*life*
and	→	*exciting*	no vertex *exciting*	—
exciting	→	*synergies*	no vertices *exciting* or *synergies*	—



Input and Output Requirement:

Your function will be called in Main.java in the following format:

```
public static void main(String[] args) throws IOException {
    final GraphPoet nimoy = new GraphPoet(new File( pathname: "src/assignment3/corpus.txt"));
    System.out.println(nimoy.poem(new File( pathname: "src/assignment3/input.txt")));
}
```

Input 1:

The **constructor** of GraphPoet will take in a file called “corpus.txt” with all the bridge words. Create a Weighted Graph of these words.

Example of “corpus.txt”

To explore strange new worlds
To seek out new life and new civilizations

Input 2:

In GraphPoet, there is a method named poem that takes in an input without the bridge words.

Example of “input.txt”

➔ *Seek to explore new and exciting synergies!*

Output:

Print to the console the final string that is made:

Example using “corpus.txt” and “input.txt”

➔ Seek to explore **strange** new **life** and exciting synergies!

Weighted Graph:

What if more than one word followed the previous?

We create a bigram model where we keep track of the weights of each word.

Example:

Input 1:

> *To explore happy new worlds*
> *To seek out new life and new civilizations*
>
> *To explore strange new things*
>
> *To explore happy new things*

Input 2:

➔ *Seek to explore new and exciting synergies!*

- *Notice there is two types of bridge words between ‘explore’ and ‘new’:*
  Explore -> happy -> new (2) Appears twice in the paragraph above

Explore -> strange -> new (1) Appears once in the paragraph above.

As a result, there will be two edges between ‘explore’ and ‘new’

‘happy’ with weight 2

‘strange’ with weight 1

Return an output with the highest weight (i.e. weight 2):

Output:

➔ *Seek to explore **happy** new life and exciting synergies*

Suggested Implementation

You are given ‘Main.java’ and ‘GraphPoet.java’

Do not modify the name of the file or the given methods. You can choose to add any additional methods or classes to help solve your problem.

1. Constructing a Graph with Vertex.java

```
class Vertex<L>{
        private L name;
        private Map<L, Integer> edges; // L is a vertex, Integer is the weight
}
```

2. For every word in the corpus, create a vertex and a map of the word following it (bigram).

3. Take in the poem input and use the graph to find bridge words with the **max** weight.

Bonus

Web scrape a UT professor's web page, and attempt to generate a poem using their biography.

http://www.ece.utexas.edu/people/faculty/edison-thomaz

Example, scrape the above website:

Scraped Input:

Prof. Edison Thomaz is a Research Assistant Professor in the Department of Electrical and Computer Engineering at The University of Texas at Austin.

Sample Input:

"Prof Thomaz is a fantastic professor; engineering the highest levels of wearable sensors.

Sample Output:

Prof. Edison Thomaz is a fantastic professor engineering at the highest levels of wearable sensors.

Submission Requirements

Name of zip file: `Assignment3_UTEID.zip` (.gzip of .gz are also ok). Make sure that the structure of the final ZIP file is as follows:

```
Assignment3_UTEID/
      src/
      assignment3/
            Main.java
            GraphPoet.java
            corpus.txt
            input.txt
            …
```

Zip your src folder and your other files together. Then rename that ZIP to `Assignment3_UTEID.zip`

Additional Considerations

External Code: you are permitted to use any classes or interfaces within the java.lang, java.io, and java.util standard packages. You are specifically prohibited from making use of another student's code (including students who may have taken the class in previous semesters).

Understandability: Comment your program so that its logic would be readily apparent to any software engineer who is familiar with standard data structures and algorithms.

Re-use: Design your code so it is suitable for future adaptations and/or expansion.

Warning: You may not acquire, from any source (e.g., another student or an internet site), a partial or

complete solution to this problem. You may not show another student your solution to this assignment. You may not have another person (TA, current student, former student, tutor, friend, anyone) “walk you through” how to solve this assignment. Review the class policy on cheating from the syllabus.