Programming Assignment 3

Nathan Friend

COM S 535: Algorithms for Large Data Sets:
Theory and Practice

2018/11/16

Data structure used to implement `Weighted Q`:

Internally, my `WeightedQ` class uses a simple `ArrayList<WeightedItem>`. The `WeightedItem` class is a simple container structure that holds the item (a `String` link, in this project) and the item's weight. This `WeightedItem` class implements the `Comparable` interface, which allows the items to be sorted (by weight) using the `Collections.sort()` method.

When the `add()` method is called, items are simply appended to the end of the internal list, regardless of their weight. When the `extract()` method is called, the `WeightedQ` class first sorts the list and then removes and returns the first item in the list. This allows `add()`s to be very fast, which is important for this use case, since `add()` is called many more times than `extract()`.

Pseudo code of your crawling algorithm. Please describe how your procedure will ensure that the graph formed will have exactly `max` many vertices:

```
// variable declarations/initializations
Create a list of edges that will eventually be written to fileName;
Create a new WeightedQ instance (wq) and add seedUrl;
Create an int to track the number of URLs we've visisted (visitedCount);
Create a list of Strings to track which URLs we've visited (visited);

// crawling algorithm
While wq is not empty and visitedCount < max:

    String url = wq.extract();

    If we've already visited this URL, continue;
    If this URL is disallowed by robots.txt, continue;

    visitedCount++;
    visited.add(url);

    Make a GET request to get the URL's HTML;
    Extract all the links from the HTML;
    For each link:
        Add link to wq;
        Add the edge (URL -> link) to the final list of edges;

    If we've made 10 requests, pause for 2 seconds;

Finally, remove any edges from the list of all edges that include URLs
that we didn't crawl;

Write the list of edges to the file;
```

(The real implementation of this pseudo-code can be viewed in `WikiCrawler.java`, in the `crawl()` method.)

This procedure ensures that the final graph will only have `max` many vertices by:

1. ensuring that only `max` many webpages are downloaded (see the `while` loop condition), and
2. removing any edges that include webpages that weren't downloaded.

Output of the program `MyWikiRanker` (top 20 page rank, in degree and outdegree pages and Jaccard Similarities):

Seed URL: `/wiki/Cello`

Keywords: `["cello", "bow", "neck", "fingerboard", "pegbox", "scroll", "pegs", "endpin", "f-holes", "f-hole", "pizzicato", "vibrato", "thumb", "string", "strings", "rosin", "orchestra", "symphony", "Stradivarius", "Stradivari", "Yo-Yo", "Rostropovich", "Casals", "Maisky", "Isserlis", "Starker", "violoncello"]`

Output:

```
A: Top 20 links based on outdegree:
     1. /wiki/Bows_for_Musical_Instruments
     2. /wiki/Cello_Concerto_in_E_major_(Cassado-Tchaikovsky)
     3. /wiki/Scottish_National_Orchestra
     4. /wiki/String_Quartet,_Op._3_(Berg)
     5. /wiki/Symphony_No._2_(Honegger)
     6. /wiki/Ariel_String_Quartet
     7. /wiki/The_Regent_String_Quartet
     8. /wiki/Concerto_for_Double_String_Orchestra_(Tippett)
     9. /wiki/Cello_Sonata_No._1_(Reger)
    10. /wiki/National_Taiwan_Normal_University
    11. /wiki/Kutcher_String_Quartet
    12. /wiki/Albert_Augustine_Strings
    13. /wiki/Aviv_String_Quartet
    14. /wiki/London_Festival_Orchestra
    15. /wiki/Benedetto_Marcello
    16. /wiki/New_Orford_String_Quartet
    17. /wiki/Stanford_String_Quartet
    18. /wiki/Tyburn_String_Quartet
    19. /wiki/Oslo_String_Quartet
    20. /wiki/String_Quartets_(Schumann)

B: Top 20 links based on indegree:
     1. /wiki/BWV_1011
     2. /wiki/String_Quartet_No._12_(Schubert)
     3. /wiki/Frankfurt_Radio_Symphony_Orchestra
     4. /wiki/Bows_for_Musical_Instruments
     5. /wiki/Solo_Cello_Sonata_(Ligeti)
```

6. /wiki/Cello_Concerto_in_E_major_(Cassado-Tchaikovsky)
7. /wiki/Cello_Symphony
8. /wiki/Sound-hole
9. /wiki/Cello_suites
10. /wiki/String_Quartet_No._2_(Schoenberg)
11. /wiki/String_Quartet,_Op._3_(Berg)
12. /wiki/Cello_Concerto_in_D_minor_(Cassado)
13. /wiki/String_basses
14. /wiki/Cello_Rock
15. /wiki/Autopista_de_Pau_Casals
16. /wiki/String_quintets
17. /wiki/Baroque_bow
18. /wiki/Symphony_No._7_(Williamson)
19. /wiki/Electric_cello
20. /wiki/Cello_Concerto_(Carter)

C: Top 20 links based on page rank with approximation = 0.01 and teleportation = 0.85:

1. /wiki/Cello
2. /wiki/String_quartet
3. /wiki/Orchestra
4. /wiki/String_instrument
5. /wiki/Bow_(music)
6. /wiki/String_section
7. /wiki/Symphony
8. /wiki/Mstislav_Rostropovich
9. /wiki/Pizzicato
10. /wiki/Yo-Yo_Ma
11. /wiki/Fingerboard
12. /wiki/Chicago_Symphony_Orchestra
13. /wiki/Philadelphia_Orchestra
14. /wiki/Boston_Symphony_Orchestra
15. /wiki/Pablo_Casals
16. /wiki/Rosin
17. /wiki/String_orchestra
18. /wiki/Tuning_peg
19. /wiki/Orchestral_enhancement
20. /wiki/Sympathetic_string

D: Top 20 links based on page rank with approximation = 0.005 and teleporation = 0.85:

1. /wiki/Cello
2. /wiki/String_quartet
3. /wiki/Orchestra
4. /wiki/String_instrument
5. /wiki/Bow_(music)
6. /wiki/String_section
7. /wiki/Symphony
8. /wiki/Mstislav_Rostropovich
9. /wiki/Pizzicato

10. /wiki/Yo-Yo_Ma
11. /wiki/Fingerboard
12. /wiki/Chicago_Symphony_Orchestra
13. /wiki/Philadelphia_Orchestra
14. /wiki/Boston_Symphony_Orchestra
15. /wiki/Pablo_Casals
16. /wiki/Rosin
17. /wiki/String_orchestra
18. /wiki/Tuning_peg
19. /wiki/Orchestral_enhancement
20. /wiki/Sympathetic_string

E: Top 20 links based on page rank with approximation = 0.001 and teleporation = 0.85:

1. /wiki/Cello
2. /wiki/String_quartet
3. /wiki/Orchestra
4. /wiki/String_instrument
5. /wiki/Bow_(music)
6. /wiki/String_section
7. /wiki/Symphony
8. /wiki/Mstislav_Rostropovich
9. /wiki/Pizzicato
10. /wiki/Yo-Yo_Ma
11. /wiki/Fingerboard
12. /wiki/Chicago_Symphony_Orchestra
13. /wiki/Philadelphia_Orchestra
14. /wiki/Boston_Symphony_Orchestra
15. /wiki/Pablo_Casals
16. /wiki/Rosin
17. /wiki/String_orchestra
18. /wiki/Tuning_peg
19. /wiki/Orchestral_enhancement
20. /wiki/Sympathetic_string

Exact Jaccard similarity between lists A and B: 0.0811
Exact Jaccard similarity between lists A and C: 0.0000
Exact Jaccard similarity between lists A and D: 0.0000
Exact Jaccard similarity between lists A and E: 0.0000
Exact Jaccard similarity between lists B and C: 0.0000
Exact Jaccard similarity between lists B and D: 0.0000
Exact Jaccard similarity between lists B and E: 0.0000
Exact Jaccard similarity between lists C and D: 1.0000
Exact Jaccard similarity between lists C and E: 1.0000
Exact Jaccard similarity between lists D and E: 1.0000

Number of iterations for your page rank algorithm to converge (within ε) on the graph `wikiTennis.txt`, for all three choices of epsilon, when β = 0.85 and β = 0.25:

Epsilon (ε)	Teleportation (β)	Number of iterations
0.01	0.85	25
0.005	0.85	29
0.001	0.85	39
0.01	0.25	5
0.005	0.25	5
0.001	0.25	6

A note on documentation: the documentation for this programming assignment can be found in Javadoc format in the `/doc` directory.