Extraction and analysis of subgraphs from online networks

Goals:

- Extract networks from online sources like https://dblp.org/.
- Given a collection of author IDs, there's an edge between a pair if they have a common paper, which can be identified by extracting the lists and looking for intersections.
- Identify statistical properties that are common or different across such networks

Technologies: Java, REST, JAXB, HTML, JavaScript, CSS
Software: Spring Boot

Source code repository:
https://github.com/Rohith-India/CS3055/tree/main/Networks/graph_sb

How to build and start the server using java command:

./gradlew clean build -x test (build)

To generate graph online (for the selected authors):
java -jar build/libs/graph_sb-0.0.1-SNAPSHOT.jar (to start server)
http://localhost:8080/graph.html (UI)

To generate graph for all 3.22 million authors from offline database:
java -Xmx4g -cp ".;libs/*;build/libs/*" com.iith.graph_sb.controller.GraphController (offline)

To generate statistics:
java -Xmx4g -cp ".;libs/*:build/libs/*" com.iith.graph_sb.graph.Graph6 (offline)

Application Programming Interfaces (APIs):

1. /findGraph - find common links/edges among authors provided in input

```
@PostMapping("/find")
public ResponseEntity<Object> findGrah(@RequestBody String[] names) {
```

- The server reads the contents of Persons.csv file and maintains a Map of name, id of all persons (One time activity at the time of start of the server)
- The API reads the id corresponding to each name in the input
- The API builds a URL http://dblp.org/<pid>.xml for each person name in the request and gets the XML response using URLConnection
- The XML response is converted into a java object called Dblpperson using JAXB
- Populate all the publications for each person into a Map
- Find combination of coauthors from the list of all the persons

Request:
curl -H "Content-Type: application/json" -X POST -d "[\"Puyan Mojabi\", \"Diana Chirkova\", \"Jim Gray 0001\", \"Ani Thakar\", \"Peter Z. Kunszt\"]" http://localhost:8080/find

Response:

```
{
  "nodes": [
    {
      "name": "Puyan Mojabi",
      "id": 1
    },
    {
      "name": "Diana Chirkova",
      "id": 2
    },
    {
      "name": "Jim Gray 0001",
      "id": 3
    },
    {
      "name": "Ani Thakar",
      "id": 4
    },
    {
      "name": "Peter Z. Kunszt",
      "id": 5
    }
  ],
  "links": [
    {
      "source": 1,
      "target": 2
    },
    {
      "source": 3,
      "target": 4
    },
    {
      "source": 3,
      "target": 5
    },
    {
      "source": 4,
```

```
    "target": 5
   }
  ]
}
```

2. /search/{name} - Search for all the matching person names in dblp for a given search string.

```
@GetMapping("/search/{name}")
  public List<String> searchMatchingNames(@PathVariable("name") String name) {
```

Request:
curl -X GET -H "Content-Type: application/json" http://localhost:8080/search/Jim%20Gray

Response:
["Jim Gray","Jim Gray 0001","Jim Gray 0002"]

Note: When user specify '*' for name, the API returns all authors in the Dblp system

Some of the Dblp statistics are collected by running the following methods from main:

3. generateAuthorsGraphFromOnlineDblpDB()
   Request: The names of authors are extracted from input.txt
   Here is the sample input:
   Puyan Mojabi
   Diana Chirkova
   Jim Gray 0001
   Ani Thakar
   Peter Z. Kunszt

   Response: The co-authors combination would be extracted into Output.csv file
   Here is the sample response:
   1,2
   3,4
   3,5
   4,5

4. generateAuthorsGraphFromOfflineDblpDB()
   Input: dblp.xml.gz
   Output: AuthorGraph.csv (Here are first few lines of 20731612 records):
   825907,2637681
   3121656,3185131
   2557081,2944882
   782753,618539
   1751281,476910

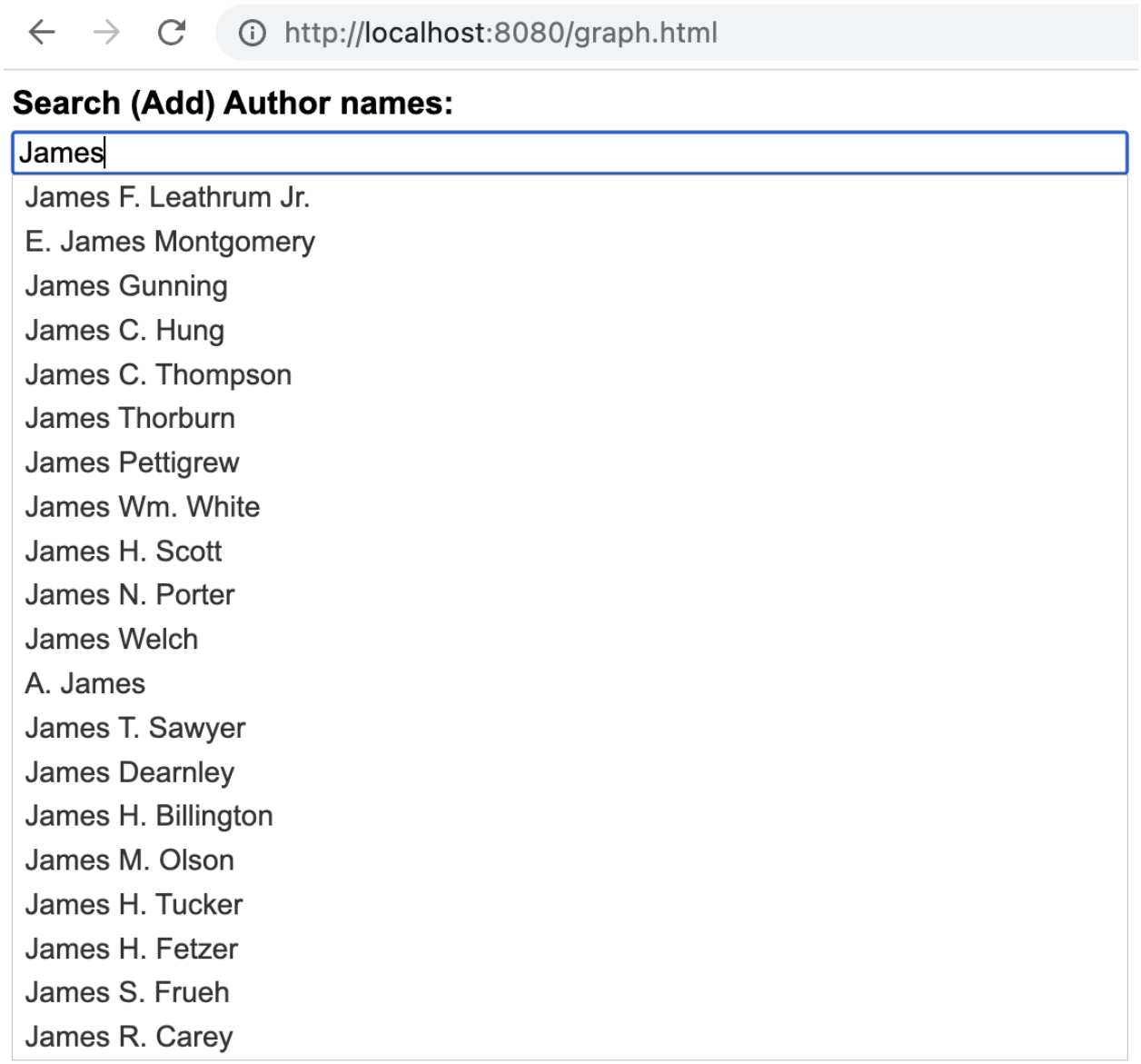
3099801,3131679
1802389,2169692
126976,2875815
3053905,1296117
3019247,2925349
207377,440693

Note: It takes about 6 hours to generate the graph for the complete offline database

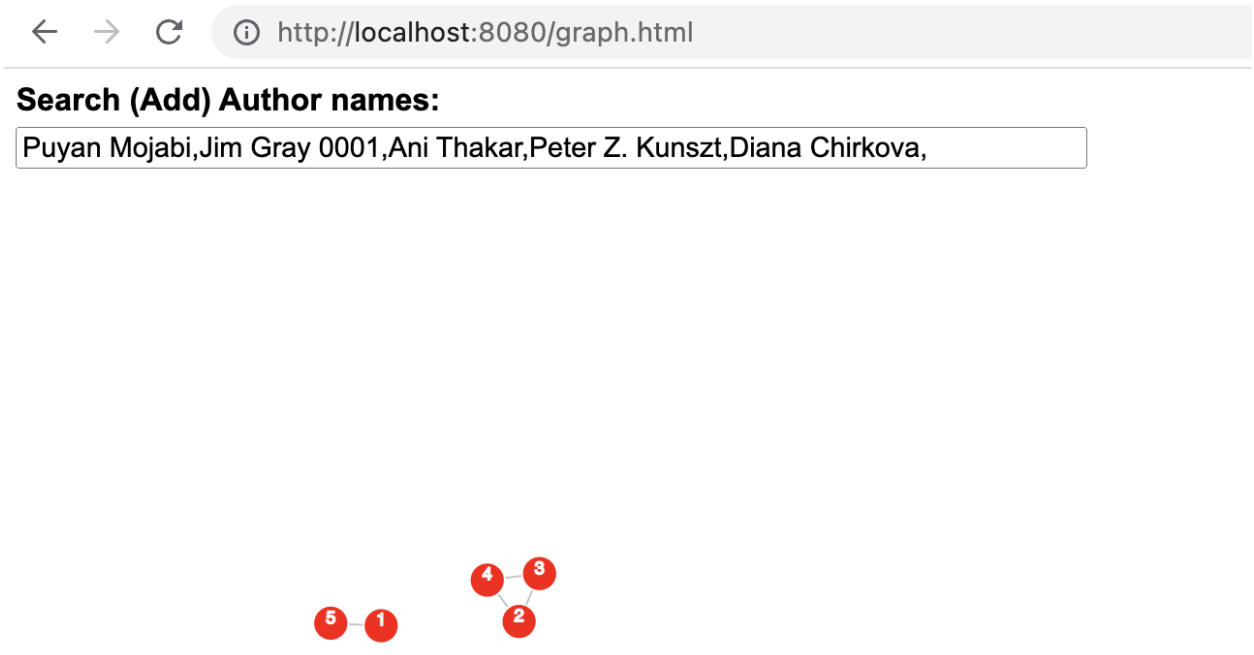
5. generatePublicationsCountsPerYearFromOfflineDblpDB()
   Input: dblp.xml.gz
   Output: (Here are the last few lines of output)
   2015 300926
   2016 313061
   2017 337361
   2018 371893
   2019 414700
   2020 429754
   2021 451607
   2022 442628
   2023 78783

<u>UI Screens:</u>

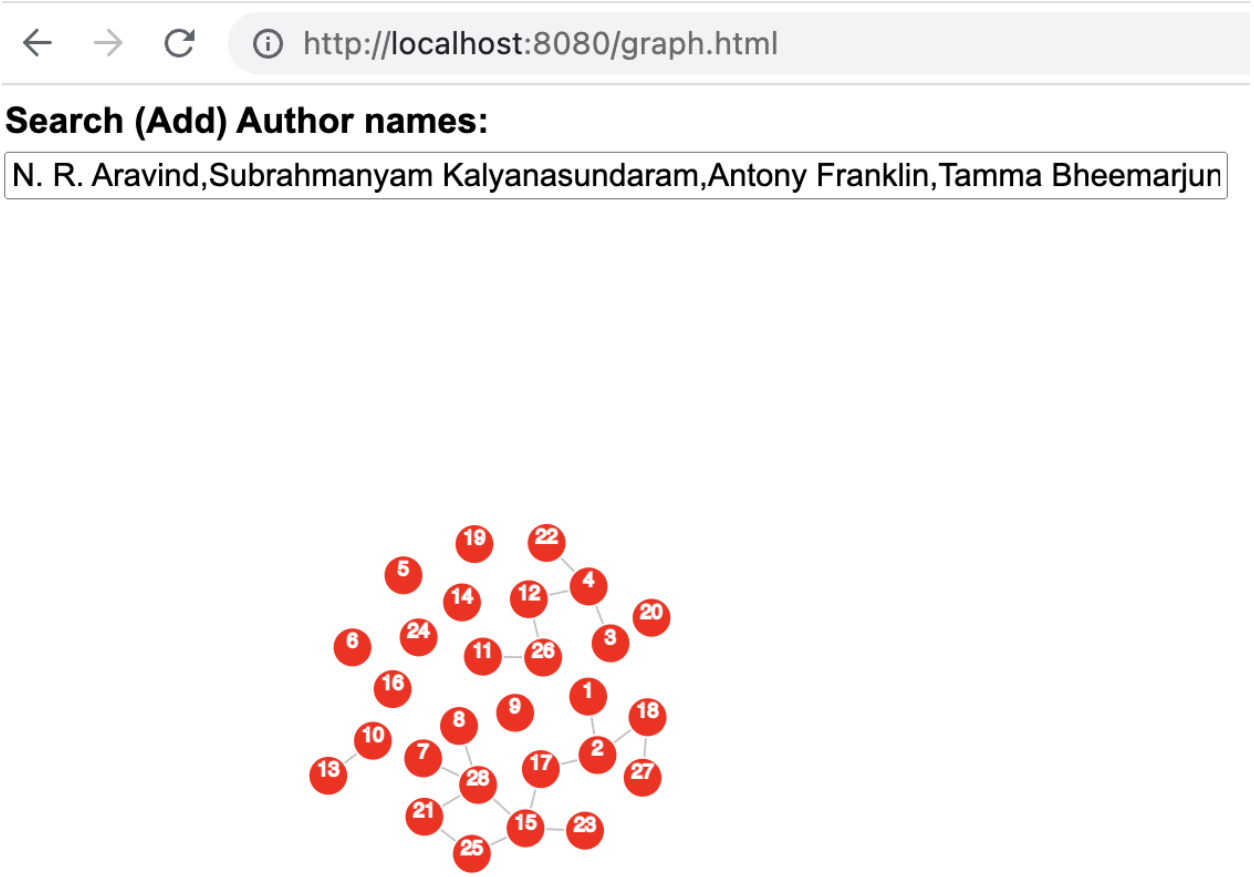
1. UI screen to select the authors for which graph needs to be generated (using AutoComplete feature):


http://localhost:8080/graph.html
Search (Add) Author names:
James
James F. Leathrum Jr.
E. James Montgomery
James Gunning
James C. Hung
James C. Thompson
James Thorburn
James Pettigrew
James Wm. White
James H. Scott
James N. Porter
James Welch
A. James
James T. Sawyer
James Dearnley
James H. Billington
James M. Olson
James H. Tucker
James H. Fetzer
James S. Frueh
James R. Carey

2. Graph: Once all the author names are entered in the above screen, the graph would be generated on pressing the ENTER key



N. R. Aravind,Subrahmanyam Kalyanasundaram,Antony Franklin,Tamma Bheemarjuna Reddy,Jyothi Vedurada,Saketha Nath Jagaralpudi,Saketha Nath Jagarlapudi,Sakethanath Nath Jagarlapudi,J. Saketha Nath,Karteek Sreenivasaiah,Maria Francis,Praveen Tammana,Nitin Saurabh,Rajesh Kedia,Ramakrishna Upadrasta,Rameshwar Pratap,Sathya Peri,Rogers Mathew,Manish Singh,Sobhan Babu Chintapalli,P. K. Srijith,C. Siva Ram Murthy,Rakesh Venkat,Shirshendu Das,Maunendra Sankar Desarkar,Kotaro Kataoka,Fahad Panolan,Vineeth N. Balasubramanian,



Dblp statistics:

The end goal is to derive the following parameters for Dblp data:

- Degree of distribution
  - The degree of a node in a network is the number of connections or edges the node has to other nodes.
- Diameter of giant component
  - In network theory, a giant component is a connected component of a given random graph that contains a significant fraction of the entire graph's vertices. The diameter of such a giant component is to be calculated here.
- Average distance
  - The average distance in a graph is defined as the average length of a shortest path between two vertices, taken over all pairs of vertices.
- Clustering coefficient
  - In graph theory, a clustering coefficient is a measure of the degree to which nodes in a graph tend to cluster together.
- No of papers by Year

Dataset used for offline analysis:

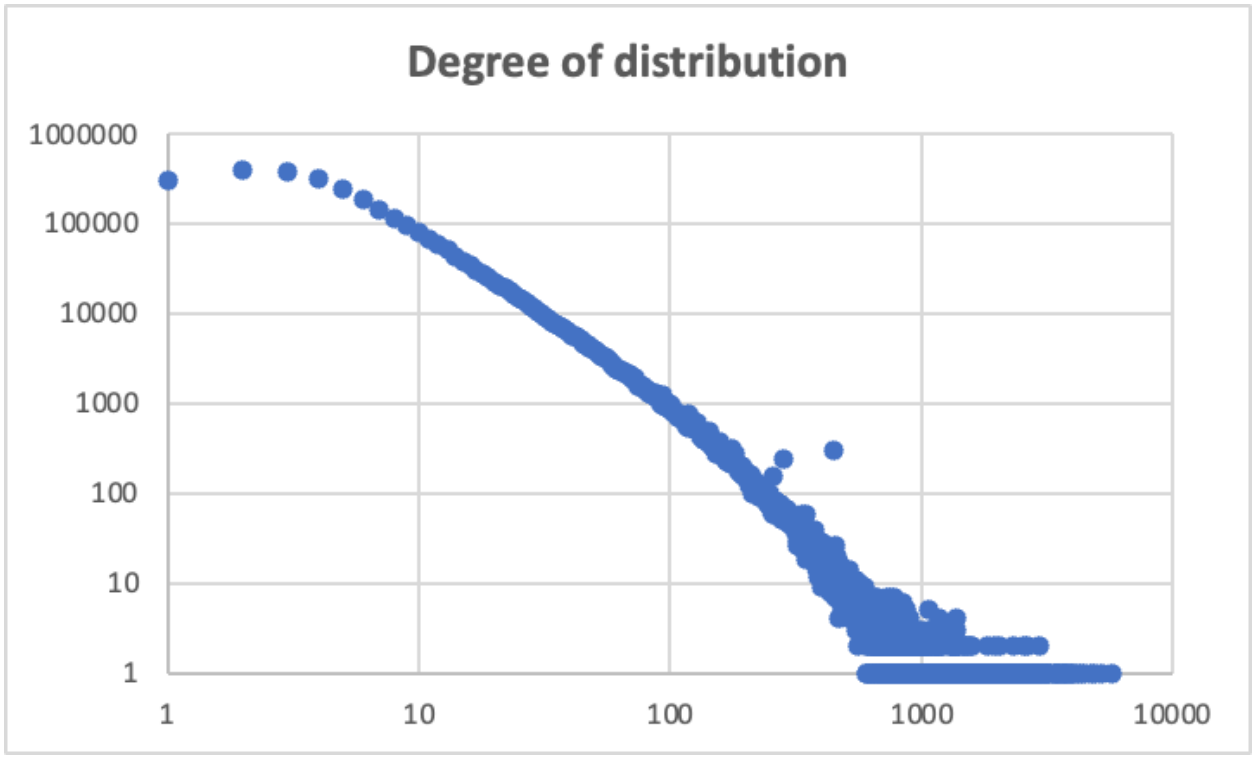
https://dblp.uni-trier.de/xml/dblp.xml.gz (2023-04-05 01:39)

Total number of authors: 3218009 (about 3.22 million)
Total number of publications: 6593055 (about 6.6 million)

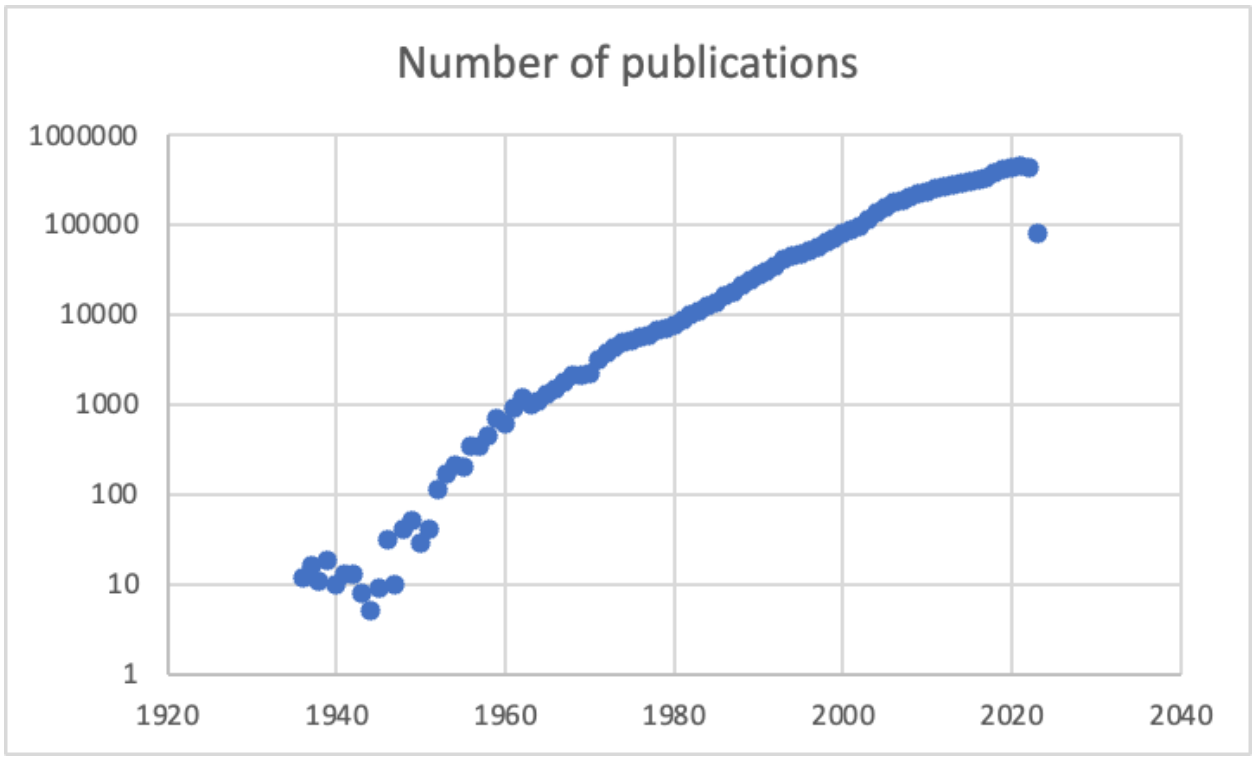
Number of vertices	***= 3088808* (about 3.1 million)***
Number of edges	***= 20731612 (about 20.73 million)***
Maximum degree	***= 5716***
Clustering coefficient	***= 0.651***
Diameter of giant component**	***= 19***
Average distance**	***= 4.31***

* There are 129201 authors who do not form any edges
****** The analysis is done for about 50000 vertices to calculate these two parameters.

Degree distribution:



No of papers by Year:



References:

https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/
https://jqueryui.com/resources/demos/autocomplete/multiple.html
https://gist.github.com/heybignick/3faf257bbbbc7743bb72310d03b86ee8
https://introcs.cs.princeton.edu/java/45graph/