Universität Trier
Fachbereich IV – Informatik
Dr. Michael Ley

Trier, den 9.7.2021

# Java-Übungsblatt zu Werkzeuge der Informatik (Praktikum)

## Sommersemester 2021

**Abgabe bis 16.8.2021 an `ley@uni-trier.de` Betreff: `Praktikum`**

Geben Sie Ihren Namen an!!! Es muß ein lauffähiges Programm und eine Text-Datei mit der Ausgabe des Programms abgegeben werden. Ergänzendes Material zu dieser Übung finden Sie auf Moodle. Verwenden Sie als Mail-Anhänge nur `.txt` und `.java` Dateien, jedoch KEINE `.tar`, `.zip`, `.jar`, `.rar`, usw. Dateien.

Sie dürfen die Aufgabe als Gruppe bearbeiten (maximal 3 Studierende). Es muß dann eine gemeinsame Abgabe erfolgen. Mehrere gleiche oder fast identische Abgaben werden als Täuschungsversuch gewertet.

Die `dblp.xml.gz`-Datei und `dblp.dtd` finden Sie in der tagesaktuellen Version unter `http://dblp.uni-trier.de/xml/` oder `http://dblp2.uni-trier.de/xml/`.

Es handelt sich wieder um die Ihnen vom C-Übungsblatt bekannte eine große xml-Datei, die alle bibliographischen Sätze (Records) des dblp-Literaturservers enthält. Diesmal muß die xml-Datei mit einem Java-SAX-Parser einlesen werden und es sollen sinnvolle Hauptspeicherdatenstrukturen mit den Klassen des Java-Collection-Frameworks (Set, List, Map, ...) aufgebaut werden. Sie können Java-8-Streams benutzen oder sich auf konventionelle Java-7-Techniken (Iteratoren etc.) beschränken. Sie können die Java-Klassen aus dem Package `saxCollections.coauthors` verwenden — müssen dies jedoch nicht.

Unter `http://dblp.uni-trier.de/xml/docu/dblpxml.pdf` finden Sie eine ausführliche Beschreibung der dblp-XML-Datei, die weit über das hinausgeht, was Sie für dieses Übungsblatt benötigen.

### Ü 1: publication stream graph — 100 Punkte

Die `dblp.xml`-Datei enthält Personen- und Publikationsrecords.

Personenrecords haben das Tag `www` und einen mit `homepages/` beginnenden Schlüssel. Aus den Personenrecords sollten Sie die Informationen über Synonyme entnehmen, also die Angabe, ob eine Person unter mehreren Namen in dblp bekannt ist.

Tipp: Sie können die Java-Klassen `PersonName` und `Person` aus dem Package `saxCollections.coauthors` (fast) unverändert verwenden. Auch der Parser muß nicht verändert werden.

In dblp sind Publikationen in Streams gruppiert. Ein Stream enthält entweder alle zu einer Zeitschrift gehörenden Artikel oder alle innerhalb einer Tagungsreihe erschienenen Veröffentlichungen. Die Schlüssel der dblp-Records sind hierarchisch aufgebaut. Sie haben die Syntax von (Unix-)Dateinamen, das Trennzeichen der Namensteile ist /. Für die Aufgabe sind nur die Publikationen mit den Schlüsselpräfixen `conf/` und `journals/` relevant. Die betrachteten Schlüssel bestehen aus drei Teilen, z.B. `conf/soca/GonzelezR18`. Die ersten beiden Teile bilden die Bezeichnung des jeweiligen Streams. Alle zu der Konferenzreihe `Service-Oriented Computing and Applications` gehörenden Publikationen haben einen Schlüssel der Form `conf/soca/...`, alle in der Zeitschrift `Datenbank-Spektrum` erschienen Artikel sind am Schlüsselpräfix `journals/dbsk/` erkennbar, usw.

Es gibt zur Zeit (Juli 2021) in dblp etwa 7000 Streams. Ermitteln Sie zunächst für jeden Stream die Anzahl der an dem Stream beteiligen Personen (Autoren oder Herausgeber).

Den Stream `journals/corr/` sollte Ihr Programm ignorieren - es handelt sich hier um eine sehr große Preprint-Sammlung. `journals/corr/` würde das Ergebnis zu sehr dominieren.

Kleine Streams mit weniger als 1000 Personen sollten ignoriert werden.

Tipp: Setzen Sie diesen Schwellwert zum Testen auf einen größeren Wert, z.B. auf 3000. Der Graph wir so erheblich kleiner und Ihr Algorithmus hat eine kürzere Laufzeit.

Betrachten Sie analog zum Koautor-Graphen einen `Co-Stream-Graphen` (CSG): Jeder Stream wird durch einen Knoten repräsentiert. Die Knoten sollten mit ihren Namen (z.B. `journals/dbsk/`) annotiert werden.

Wir könnten einen ungerichteten Graphen aufbauen, bei dem die Kanten zwischen den Knoten mit natürlichen Zahlen gewichtet sind: Eine Kante `edge(`$s_1$`, `$s_2$`, w)` zwischen den Streams $s_1$ und $s_2$ mit dem Gewicht $w$ bedeutet: $w$ Personen haben in den Streams $s_1$ und $s_2$ publiziert.

Wir betrachen jedoch eine gerichtete Variante dieses Graphens: Von jedem Stream aus wird die Kante mit dem höchsten Gewicht ausgewählt. Der Ausgangsknoten ist der aktuelle Stream, der Pfeil zeigt auf den Partner-Stream. Der Ausgangsgrad eines Knotens kann also Null (kein gemeinsamer Autor mit anderen Streams) oder Eins sein. Der Eingangsgrad kann dagegen größer sein.

Geben Sie in einer Text-Datei für die relevanten Streams die jeweils andere Seite der ausgehenden und eingehenden Kanten an.

```
conf/aaai (27734) :
  out
    conf/ijcai
  in
    conf/ijcai
    conf/uai
    conf/flairs
    conf/ictai
    conf/atal
...
conf/atal (8901) :
  out
     conf/aaai
  in
...
```

Dieser kurze Ausschnitt aus der Ausgabedatei beschreibt folgende Situation: Am Stream `conf/aaai` sind 27734 Personen beteiligt. Aus `conf/aaai` zeigt eine Kante auf `conf/ijcai`. Aus `conf/ijcai`, `conf/ucai`, `conf/flairs`, `conf/ictai` und `conf/atal` zeigt jeweils eine Kante auf `conf/aaai`, usw.