Universität Passau
Fakultät für Informatik und Mathematik

FORWISS

Bachelorarbeit: Scopus DOI Crawler

im Studiengang Internet Computing

zur Erlangung des akademischen Grades
Bachelor of Science

Thema:	Scopus DOI Crawler
Autor:	Manuel Donaubauer (donaub13@stud.uni-passau.de) MatNr. 63658
Version vom:	23. März 2015
Betreuer:	Prof. Dr. Tomas Sauer

Abstract

Die Aufgabenstellung dieser Bachelorarbeit war die Implementierung eines Computerprogramms, welches sämtliche Autorinformationen zu bestimmten Publikationen von der Scopus-Webseite beschafft. In der Ausarbeitung wird zunächst auf die Funktionsweise und Unterschiede von Web Crawlern eingegangen. Als Quellen dienten hierzu diverse wissenschaftliche Artikel, welche im Literaturverzeichnis nachgeschlagen werden können. Die darauffolgenden Kapitel beschreiben den genauen Aufbau und die Funktionsweise des entwickelten Scopus DOI Crawlers. Dabei wird zu Beginn auf die Unterschiede zu anderen Crawlern eingegangen. Eine Erklärung zur grafischen Oberfläche und den verwendeten Libraries wird ebenfalls detailliert aufgeführt. Probleme, welche während der Entwicklung aufgetaucht sind sowie ein veranschaulichendes Beispiel runden diese Ausarbeitung ab.

Inhaltsverzeichnis

Abbildungsverzeichnis **5**

Tabellenverzeichnis **6**

Listingverzeichnis **7**

Abkürzungsverzeichnis **8**

1 Einleitung **9**

2 Web Crawler **9**
2.1 Funktionsweise . . . 9
2.2 Crawler Policies . . . 10
2.3 Typen von Web Crawlern . . . 11
2.4 Probleme von Web Crawlern . . . 12

3 Scopus DOI Crawler **14**
3.1 Kurzbeschreibung . . . 14
3.2 Unterschiede zu anderen Crawlern . . . 14
3.3 Entwicklungsumgebung . . . 14
3.4 Systemvoraussetzungen . . . 14
3.5 Grafische Oberfläche . . . 15
3.6 Funktionsweise . . . 16
3.7 Verwendete Libraries . . . 20
3.8 Ordnerstruktur . . . 21
3.9 Ausgabedateien . . . 22

4 Probleme bei der Implementierung **25**
4.1 Kein direkter Datenbankzugriff . . . 25
4.2 Finale Datei hat zu viele Zeilen beziehungsweise ist zu groß . . . 25
4.3 Liste von Autoren konnte nicht richtig ausgelesen werden . . . 25
4.4 Anführungszeichen wurde nicht richtig escaped . . . 26
4.5 csv Dateien wurden nicht richtig vollständig eingelesen . . . 26
4.6 Crawlvorgang beschleunigen . . . 26
4.7 Autorinformationen von mehr als 2000 Einträgen können nicht heruntergeladen weren . . . 27
4.8 Webseite liefert Ressourcen zu langsam . . . 27
4.9 Fehler während des Vorgangs . . . 27
4.10 Bei einem Fehler konnte das Programm nicht erneut gestartet werden . . . 27
4.11 Der Speicher der JVM reicht nicht aus . . . 28
4.12 Firefox wird bei einem Fehler nicht beendet . . . 28
4.13 Seite mit den Autor-/Publikationsinformationen wird nicht vollständig geladen . . . 28

5 Beispiel **29**
5.1 Eingaben . . . 29
5.2 GUI Log Bereich . . . 29
5.3 Ausgabe: log_scheduler.txt . . . 30

5.4 Ausgabe: summary.csv 32
5.5 Ausgabe: complete_0_doiCrawler_only2csv.csv 34

6 Ausblick **36**

7 Fazit **36**

Literaturverzeichnis **37**

Anhang **38**

Eidesstattliche Erklärung **38**

Abbildungsverzeichnis

1 Visualisierung der Funktionsweise von Web Crawlern 9
2 Grafische Oberfläche des Scopus DOI Crawlers 15
3 Aufbau der DOI Input Datei 16
4 Scopus Webseite einer Publikation 18
5 Scopus Webseite eines Autors 19
6 Ordnerstruktur der Ausgabe 21

Tabellenverzeichnis

1 Splittübersicht der finalen Datei 22
3 Übersicht aller wählbaren Spalten 23
4 Zusammenfassung des gesamten Vorgangs 32
5 Finale Datei eines Vorgangs 34

Listingverzeichnis

1 Auszug aus einer Log Datei, die während eines Vorgangs angelegt wird. 30

Abkürzungsverzeichnis

CSV ..... Comma Separated Values
DOI ..... Digital Object Identifier
GUI ..... Graphical User Interface
HTML ..... Hypertext Markup Language
HTTP ..... Hypertext Transfer Protocol
JVM ..... Java Virtual Machine
URL ..... Uniform Resource Locator

1 Einleitung

Die Scopus Datenbank ist seit dem Jahr 2004 online verfügbar und umfasst über 50 Millionen wissenschaftliche Publikationen[1]. Die Datenbank lässt sich anhand von Suchbegriffen oder einer konkreten DOI durchsuchen. Detaillierte Informationen zu jeder Publikation und deren Autoren können vom Anwender abgerufen werden. Aufgrund der Tatsache, dass bestimmte Informationen nicht gebündelt heruntergeladen werden können, wurde der Scopus DOI Crawler entwickelt. Das Programm extrahiert Informationen zu Veröffentlichungen von Autoren einer bestimmten DOI und speichert diese lokal.
Dieses Dokument stellt zunächst den Aufbau und die Funktionsweise von Web Crawlern dar. In den weiteren Kapiteln wird auf den implementierten Scopus DOI Crawler genauer eingegangen.

2 Web Crawler

Große Internet Suchmaschinen wie Google, Yahoo oder Bing benötigen einen Mechanismus zum automatischen Indexieren des World Wide Webs. Dieser Vorgang wird zur Erstellung der Ergebnisse von Suchanfragen benötigt. Damit die Suchergebnisse immer aktuell sind, wird das Indexieren in bestimmten Abständen wiederholt. Ein solches Programm, dass diese Aufgabe übernimmt, wird als „Web Crawler“, „Spider“, „Ant“, „Automatic Indexer“, „Bot“ oder „Worm“ bezeichnet [MK00].

2.1 Funktionsweise

Im folgenden Kapitel wird die grundlegende Funktionsweise von Web Crawlern erläutert.

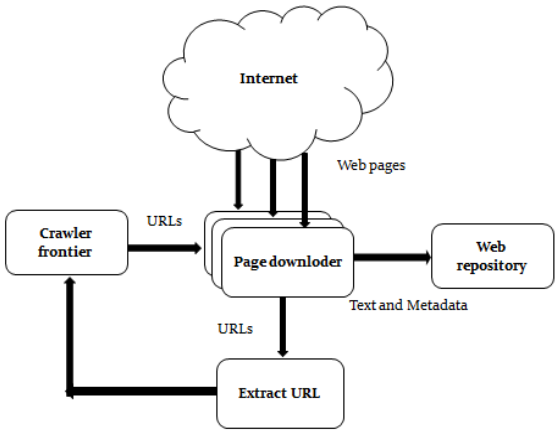

Abbildung 1: Visualisierung der Funktionsweise von Web Crawlern [TVU14]

[1]Wikipedia, vgl. `http://de.wikipedia.org/wiki/Scopus_(Datenbank)`, 02.01.2015

Die Crawler Architektur lässt sich grob in drei Hauptkomponenten einteilen. Hierbei handelt es sich um das „Web frontier“, den „Page downloader“ und das „Web repository“.

- **Crawler frontier**
  Um einen Vorgang zum Indexieren von Webseiten starten zu können, muss dem Crawler ein Startpunkt zugeteilt werden. Diese URLs werden von einem Benutzer beziehungsweise von einem anderen Programm in das Crawler frontier hinzugefügt. Das Crawler frontier beinhaltet alle Links zu Webseiten, von denen die Indexierung erstellt werden soll. Beim Starten des Vorgangs arbeitet das Programm diese Liste mit URLs, nach einer bestimmten Priorität, ab. Der Crawlvorgang endet, wenn keine URLs in der Liste des frontiers vorhanden sind oder ein Abbruch durch den Benutzer ausgeführt wird [TVU14].

- **Page downloader**
  Damit die Liste mit noch nicht besuchten URLs aus dem Crawler frontier verarbeitet werden kann, wird der Page downloader benötigt. Dieser kapselt einen HTTP Client, der eine request Anfrage an den Server, mit der zuvor ausgewählten URL, sendet. Die response beinhaltet die angefragte Webseite. Alle Links von dieser Seite werden dabei extrahiert und in das Crawler frontier kopiert. Um unnötiges warten auf große Dateien, wie Bilder oder Videos, vermeiden zu können, sollte ein passender Timeout gewählt werden [TVU14].

- **Web repository**
  Die Webseiten, die der Page downloader herunterlädt, müssen in einer Datenbank gespeichert werden. Diese Funktion übernimmt das Web repository. In dieser Datenbank werden nur reine HTML Seiten gespeichert und keine anderen Dateien, wie beispielsweise Bilder, Videos oder andere Medien. Das Web repository unterscheidet sich nur geringfügig von anderen Datenbanken oder Dateisystemen. Die für den Crawler verwendete Datenbank benötigt keinen großen Funktionsumfang. Die Hauptaufgaben sind das Speichern beziehungsweise das Updaten von vorhanden Seiten [TVU14].

2.2 Crawler Policies

Damit Web Crawler bei der Informationsbeschaffung keine Probleme bei den einzelnen Servern, beziehungsweise in Teilen von Netzwerken verursachen, sollten sie folgende policies befolgen.

- **Selection policy**
  Da ein Web Crawler immer nur einen Teil des World Wide Webs crawlen kann, müssen die für ein bestimmtes Thema relevanten Seiten, heruntergeladen werden.

Dies erfordert eine Priorisierung der einzelnen Webseiten, welche nach einem bestimmten Algorithmus durchgeführt wird [SD11].

- **Re-visit policy**
  Viele Webseiten sind sehr dynamisch. Das heißt sie werden oft bearbeitet beziehungsweise aktualisiert. So können beispielsweise Einträge hinzugefügt oder gelöscht werden. Damit der Crawler seine Datenbank immer auf dem aktuellen Stand halten kann, muss das Programm die Seiten nach einer bestimmten Zeit wieder besuchen. Falls eine Änderung stattgefunden hat, wird die Webseite erneut heruntergeladen und in der Datenbank gespeichert. Die genaue Berechnung, nach welcher Zeit welche Seite wieder besucht wird, kapselt die re-visit policy [SD11].

- **Politeness policy**
  Da Web Crawler viel schneller als menschliche Benutzer mit einer Webseite interagieren, kann dies von Performanceeinbußen bis hin zum Ausfall eines Servers führen. Durch schlecht programmierte Crawler könnten auch ganze Netzwerke überflutet und somit unter der Last zusammenbrechen, da sie kontinuierlich Bandbreite in Anspruch nehmen. Um diese Vorfälle zu vermeiden wurde die politeness policy eingeführt [SD11].

- **Parallelization policy**
  Um den das crawlen möglichst schnell durchzuführen, werden mehrere Vorgänge gleichzeitig gestartet. Die Parallelisierung kann dazu führen, dass eine Webseite von mehreren Vorgängen gecrawlt wird und somit Duplikate in der Datenbank auftauchen. Dies verschwendet die Ressourcen des Programms und führt zu Inkonsistenzen in der Datenbank. Die parallelization policy sorgt für eine fehlerfreie parallele Durchführung [SD11].

2.3 Typen von Web Crawlern

Web Crawler lassen sich in unterschiedliche Typen einteilen. Die folgende Aufzählung soll eine kurzen Überblick liefern.

- **Focused Web Crawler**
  Dieser Web Crawler sucht nach Webseiten, welche zu einem bestimmten Thema passen. Dabei bestimmt das Programm die Relevanz der aktuellen Webseite und entscheidet danach ob auf dieser Seite weiter gecrawlt werden soll oder nicht. Ein Vorteil dieser Vorgehensweise ist, dass die Anforderungen an die Hardware und das Netzwerk nicht sonderlich groß sind. Das Ergebnis ist trotz des fokussierten Vorgangs riesig [TVU14].

- **Incremental Crawler**
  Im Gegensatz zu handelsüblichen Crawlern, welche ihre Datenbank immer im glei-

chen zeitlichen Abstand aktualisieren, richtet sich der Updatezeitraum bei einem inkrementellen Crawler nach der Änderungshäufigkeit einer Webseite. Dadurch werden statische Seiten in nur sehr langen und Informationsportale, bei denen sich täglich Änderungen ergeben, in sehr kurzen Zeitabständen aktualisiert. Bei diesem Vorgang können auch weniger wichtige Seiten durch neue, bessere ersetzt werden [TVU14].

- **Distributed Crawler**
  Beim distributed crawling handelt es sich um mehrere einzelne Crawler, welche durch einen zentralen Server synchronisiert werden. Dieser sorgt für eine Kommunikation zwischen den einzelnen Programmen und wird aufgrund der geographischen Trennung der Crawler benötigt. Es wird üblicherweise der Page rank Algorithmus verwendet. Durch die Arbeitsteilung des eigentlichen Vorgangs, ist diese Vorgehensweise für Ausfälle weniger anfällig. Durch den modularen Aufbau kann das System durch andere Crawler und Komponenten erweitert werden [TVU14].

- **Parallel Crawler**
  Ein Crawler, welcher mehrere Crawling Prozesse parallel ausführt, wird als Parallel Crawler bezeichnet. Diese Vorgehensweise liefert eine erhebliche Beschleunigung im Gegensatz zu Crawlern die linear arbeiten [TVU14].

2.4 Probleme von Web Crawlern

Folgende Probleme können Web Crawler im World Wide Web verursachen.

- **Zugriff auf sensible Daten**
  Dadurch, dass ein Crawler möglichst viele Einträge und Informationen indexieren will, müssen sensible Daten davor geschützt werden. Dies betrifft alle Webseiten auf denen sich der Benutzer zuvor autorisieren muss, um einen bestimmten Inhalt abrufen zu können. Oft sind solche Inhalte auch nur gegen Bezahlung abrufbar. Falls ein Crawler nicht ausgesperrt wird, könnte dies das Geschäftsmodell der Webseite schädigen [MK00].

- **Serverüberlastung**
  Wenn ein Web Crawler eine Webseite indexiert, wird dadurch Leistung und Bandbreite eines Servers in Anspruch genommen. Falls mehrere Crawler auf einen Server zugreifen, kann es zu einer erheblichen Verlangsamung der Zurverfügungstellung der Daten kommen. Dies kann bis zum Ausfall des Servers führen. Aus diesem Grund wurde der *robots exclusion standard* eingeführt, der ausgewählte Crawlvogänge blockiert. Web Crawler sind nicht immer die Ursache für Serverausfälle. Eine zu große Anzahl an gleichzeitigen Benutzern kann ebenfalls zu Ausfällen führen [MK00].

- **Häufiges Updaten einer Webseite**
  Webseiten, wie Focus oder Stern, werden mehrfach in der Stunde aktualisiert. Dies kann zu veralteten Informationen in der Datenbank des Crawlers führen. Um solche veraltete Einträge zu vermeiden, muss die Webseite mehrmals am Tag neu indexiert werden [MK00]. Google wirkt dem Problem durch mehrfaches besuchen/indexieren am Tag eines neuen Artikels entgegen. Des weiteren sollten die technischen guidelines großer Suchmaschinen eingehalten werden[2].

[2]Google Support, vgl. `https://support.google.com/news/publisher/answer/40392?hl=en`, 13.01.2015

3 Scopus DOI Crawler

3.1 Kurzbeschreibung

Der Scopus DOI Crawler sammelt alle gewünschten Informationen zu den Autoren bestimmter Publikationen, welche der Anwender vor jedem Crawlvorgang festlegen kann. Das Programm navigiert nach dem Starten zu der gewünschten DOI, lädt alle Informationen zu allen Publikationen von jedem Autor herunter und fasst diese zu einer Datei zusammen. Dies geschieht mit allen gewünschten DOIs. Wenn alle abgearbeitet wurden, fasst der Crawler, wenn die Checkbox zum Aggregieren markiert ist, die zusammengefassten Dokumente nochmals zu einer Datei zusammen.

3.2 Unterschiede zu anderen Crawlern

Der Scopus DOI Crawler hat als Startpunkt keine Liste mit URLs sondern DOIs. Die Liste mit den DOIs bleibt während des gesamten Vorgangs unverändert. Das Programm navigiert immer mit der gleichen Vorgehensweise auf die Autorseite und lädt die Informationen von Scopus herunter. Somit crawlt das Programm nicht einen Teil des World Wide Webs sondern nur die Scopus Webseite. Dies geschieht parallel mit bis zu drei Prozessen. Die geladenen Informationen werden nicht in einer Datenbank, sondern in einer csv Dateien gesichert. Ein automatisches Aktualisieren der heruntergeladenen Daten wird nicht durchgeführt.

3.3 Entwicklungsumgebung

Der Scopus DOI Crawler wurde in der Programmiersprache Java (Version 8 Update 25) programmiert. Für das Implementieren wurde Eclipse IDE for Java Developers in der Version Luna Service Release 1 (4.4.1) verwendet. Die Entwicklung fand auf einem MacBook Pro Retina Mitte 2012 statt. Sämtliche Tests, während sowie nach der Programmierphase, wurden sowohl auf einem Windows als auch OS X Rechnern durchgeführt.

3.4 Systemvoraussetzungen

Als minimale Systemanforderungen auf Seiten des Endanwenders wird folgendes benötigt.

- Windows, Linux, Mac OS
- 4 GB Arbeitsspeicher
- Intel Core i5 oder vergleichbarer AMD Prozessor
- Java Version 8 Update 25

- Firefox (34.0.5)
- Breitband Internetanbindung

Eine Installation des Programms ist nicht notwendig. Die Datei „Crawler.jar“ muss lediglich durch einen Doppelklick gestartet werden. Eine vollständige Java und Firefox Installation setzt der Scopus DOI Crawler voraus.

3.5 Grafische Oberfläche

Der Crawler bietet eine logisch aufgebaute grafische Oberfläche, die dem Benutzer ermöglicht, die wichtigsten Einstellungen und Angaben zu tätigen.

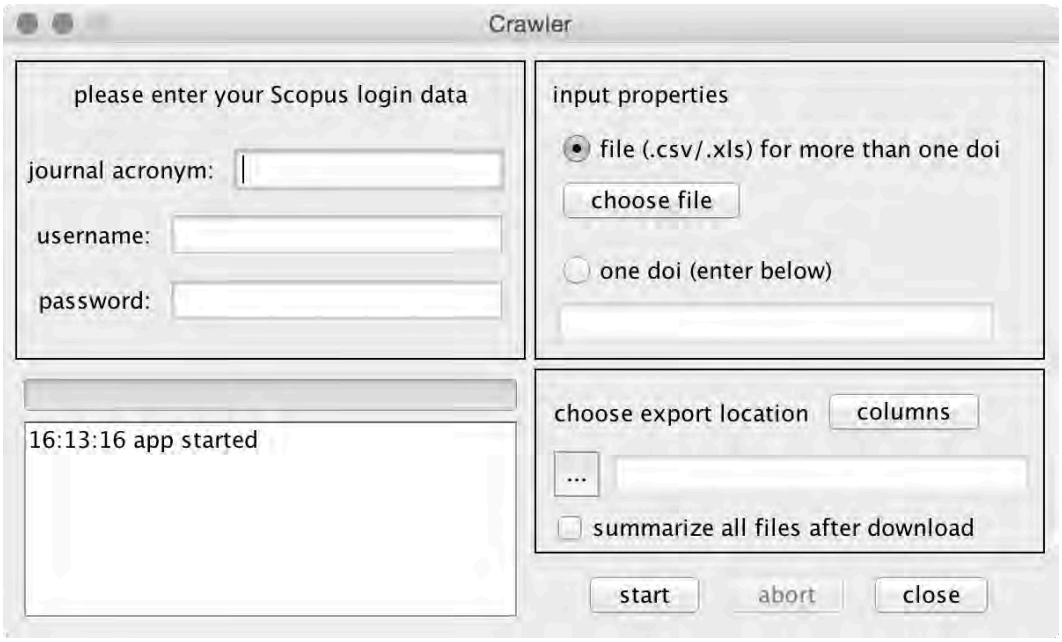

Abbildung 2: Grafische Oberfläche des DOI Crawlers

Im linken obereren Bereich der GUI muss der Benutzer die Zugangsdaten, welche er für den Crawlvorgang verwenden möchte, eingeben. 30-Tage Test-Accounts werden vom Programm ebenfalls unterstützt.
Der rechte obere Bereich des Fensters dient den Input Einstellungen. Das Programm kann nach mehreren DOIs in einem Vorgang oder nach nur einer DOI crawlen. Falls nach mehreren DOIs gesucht werden soll, ist darauf zu achten, dass die Datei die Endung xls (Excel Datei) oder csv (Komma separierte Werte) sein muss.

Falls mehrere DOIs ausgewählt wurden, sollte darauf geachtet werden, dass sich keine Duplikate in der Liste befinden. Bei einer Excel Datei muss sich die Tabelle mit den DOIs auf dem ersten Sheet befinden. Tabellennamen werden nicht herausgefiltert und sollten vermieden werden. Der Dateibrowser für das Wählen der Datei öffnet sich durch klicken auf den „choose file“ Button. Ein Dateifilter, damit nur verwendbare Dateien gewählt werden können, wurde ebenfalls implementiert.

10.1371/journal.pone.0000202
10.1371/journal.pone.0000203
10.1371/journal.pone.0000204
10.1371/journal.pone.0000205
10.1371/journal.pone.0000206

```
10.1371/journal.pone.0000202
10.1371/journal.pone.0000201
10.1371/journal.pone.0000203
10.1371/journal.pone.0000204
10.1371/journal.pone.0000205
10.1371/journal.pone.0000206
```

Abbildung 3: links: Aufbau einer xls Datei; rechts: Aufbau einer csv Datei

Der Bereich zum Auswählen des Speicherortes der heruntergeladenen und bearbeiteten Dateien befindet sich unterhalb des Import Bereichs. Durch klicken auf „...“ öffnet sich der Dateibrowser in dem der Speicherort gewählt wird. Der Button „columns“ öffnet ein Fenster, in dem der Anwender die gewünschten Spalten der finalen Dateien auswählen kann. Je nach Anzahl der Spalten splittet der Crawler die fertige csv, um zu große Dateien zu vermeiden. Dazu muss allerdings die Checkbox „summarize all files after download“ markiert worden sein. Die Anzahl an Input DOIs beträgt bei markierter Checkbox 200. Bei nicht gewünschtem Zusammenfassen der Daten können bis zu 5000 DOIs in einem Vorgang gecrawlt werden.
Im rechten unteren Bereich befinden sich drei Buttons. Der „start“ Button startet den Crawlvorgang. Der „abort“ Button sendet den Abbruch Befehl an den Crawler, welcher die bereits gestarteten DOIs noch fertig crawlt und anschließend abbricht. Der „close“ Button beendet das Programm.
Unterhalb des Login Bereichs informiert der Crawler den Benutzer über seine aktuellen Aktivitäten. Dazu gehört auch ein Fortschrittsbalken. Im Log Bereich werden Informationen wie Fehlermeldungen, Erfolgsmeldungen und Statusmeldungen ausgegeben. Am Ende eines durchgeführten Crawlvorgangs wird dem Anwender eine Zusammenfassung angezeigt.

3.6 Funktionsweise

Nach dem Start des Programms gibt der Anwender seine Scopus Login Daten im linken oberen Bereich der GUI ein. Anschließend wird eine oder mehrere DOIs, nach welchen die Autorinformationen gecrawlt werden sollen, ausgewählt. Nach dem Wählen des Dateispeicherortes und der gewünschten Spalten, wird der Vorgang gestartet.
Das Programm legt zu Beginn einen neuen Ordner für den aktuellen Vorgang an. Dabei wird der Name der DOI Listendatei oder bei nur einer DOI der Name dieser verwendet. Im nächsten Schritt startet der Firefox Browser automatisch und ruft die Scopus Login Webseite auf. Nach erfolgreicher automatischer Eingabe der Benutzerdaten überprüft das Programm, ob der Login des Benutzers erfolgreich war oder nicht. Dies geschieht anhand des Titels der Webseite, die nach dem Submit des Formulars der Startseite von

Scopus zurückgegeben wird. Im Falle eines fehlerhaften Logins, bricht der Crawler den Vorgang nach einem bestimmten Timeout ab und informiert den Anwender über den fehlerhaften Loginversuch im Log Bereich.
Bei erfolgreichem Login wird im nächsten Schritt die Liste mit den DOIs beziehungsweise die einzelne DOI eingelesen. Mit einer if Abfrage wird zuerst überprüft, ob eine Zusammenfassung der Daten am Ende gewünscht wird oder nicht. Falls es sich um mehr als 200 beziehungsweise 5000 Elemente handelt oder Duplikate in der Liste vorkommen, meldet der Crawler den Fehler in dem Log Bereich und stoppt den Vorgang. Falls alles richtig eingelesen wurde, wird ein neuer Ordner „temp“, im zuvor erstellten Ordner des aktuellen Crawlvorgangs, angelegt. Dieser dient zum temporären speichern der Dateien, die von Scopus heruntergeladen werden. Dabei wird, im späteren Verlauf, für jede DOI ein Unterordner angelegt, in dem die aktuell heruntergeladene csv Datei des aktuellen Autors gespeichert wird.
Im nächsten Schritt werden maximal drei gleichzeitige Crawlvorgänge gestartet. Dies dient der Beschleunigung der Abarbeitung von den gewünschten DOIs. Die benötigte Zeit verkürzt sich somit um knapp zwei Drittel jener Zeit, die bei fehlender Parallelisierung anfallen würde. Der eigentliche Vorgang startet mit dem Erstellen eines neuen Verzeichnisses im „temp“ Ordner. Dieses bekommt den Namen der aktuellen DOI und dient, wie oben bereits beschrieben, der temporären Speicherung von den Autor csv Dateien, die von Scopus heruntergeladen werden. Ein weiterer Ordner ebenfalls mit den Namen der aktuellen DOI wird im Verzeichnis des aktuellen Crawlvorgangs angelegt. In diesem werden zu einem späteren Zeitpunkt die fertigen Dateien abgespeichert.
Anschließend wird ein neues Firefox Profil, in dem der Dateispeicherort für die heruntergeladenen csv Dateien der Autoren festgelegt wird, erstellt. Das neue Firefox Fenster ruft die Scopus Login Seite auf und verwendet die zuvor getesteten Anmeldedaten zum Einloggen in die Datenbank. Bei erfolgreicher Weiterleitung zur Suchmaske von Scopus wählt der Crawler die Sucheigenschaft „DOI“ aus und gibt die DOI in das Suchfeld ein. Dadurch, dass die DOI immer eindeutig ist und somit nur eine Publikation in der Ergebnisliste erscheinen kann, wählt der Crawler, nach dem erfolgreichen submit des Formulars, den ersten und auch einzigen Eintrag in der Liste aus.
Die Webseite, die nach dem Klick auf den Namen der Veröffentlichung geladen wird, bietet eine Übersicht von Informationen über die gewählte Publikation (Abbildung 4). Dazu gehören zum Beispiel Titel, Veröffentlichungsdatum, Abstract, ähnliche Dokumente, Autor Schlüsselwörter und eine Liste von Autoren der gewählten Publikation. Letztere ist für den Crawlvorgang essentiell. Alle Autoren der Veröffentlichung, die das Programm in eine Liste speichert, wird aufgerufen und die Exportfunktion von Scopus für alle Publikationen des gewählten Autors verwendet. Die Datei enthält sämtliche Informationen zu allen Veröffentlichungen des gewählten Autors (Abbildung 5). Dabei wird als Dateiformat csv verwendet und in dem temporären Ordner, welcher zuvor

angelegt worden ist, heruntergeladen.

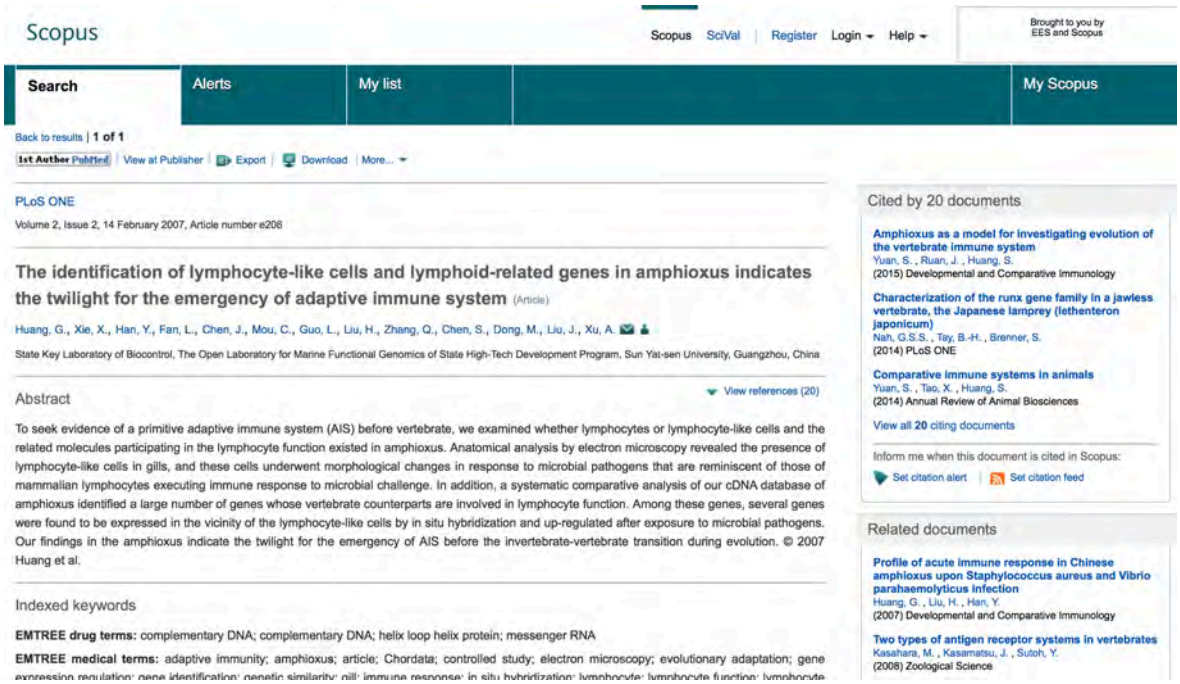

Abbildung 4: Scopus Webseite einer Publikation[3]

Nachdem das Programm auf den vollständigen Download gewartet hat, wird die heruntergeladene Datei (scopus.csv) eingelesen und zwei Spalten (crawler_doi, crawler_position) hinzugefügt. Die crawler_doi entspricht der aktuellen DOI von der die Autor Informationen gecrawlt worden sind. Die crawler_position steht für die Position des Autors in der aktuellen DOI. Nach diesem Schritt wird die geänderte csv Datei in den finalen Ordner im Crawler Arbeitsverzeichnis, unter dem Namen der aktuellen DOI wie bereits erwähnt, abgespeichert. Dieser Vorgang geschieht für alle Autoren der gewählten DOI. Falls jedoch ein Autor mehr als 2000 Veröffentlichungen hat, kann die Datei nicht heruntergeladen werden. In diesem Fall wird der Benutzer im Log Bereich darüber informiert und das Programm fährt mit dem nächsten Autor fort.
Falls es keine Fehler gab und alle Autor Dateien gespeichert wurden, werden die csv Dateien von allen Beteiligten eingelesen und in eine Datei zusammengefasst. Diese Datei beinhaltet die zwei hinzugefügten Spalten und alle Informationen zu allen Publikationen aller Autoren der gewählten DOI in der Reihenfolge der Autorposition.
Dieser Vorgang wird für alle DOIs in der zu Beginn gewählten Liste oder, falls nur eine gewählt wurde, nur für die eine angewandt. Falls Fehler bei einem Vorgang auftauchen,

[3]Scopus Publikation 10.1371/journal.pone.0000206, vgl. `http://www.scopus.com.scopeesprx.elsevier.com/record/display.url?eid=2-s2.0-46449129564&origin=resultslist&sort=plf-f&src=s&st1=10.1371%2fjournal.pone.0000206&sid=25E12A6DC689F93F4C5060E9EDFB5700.euC1gMODexYlPkQec4u1Q%3a20&sot=b&sdt=b&sl=33&s=DOI%2810.1371%2fjournal.pone.0000206%29&relpos=0&relpos=0&citeCnt=20&searchTerm=DOI%2810.1371%2Fjournal.pone.0000206%29`, 06.02.2015

bricht das Programm ab und startet zwei bis drei weitere Versuche. Da der zu Beginn angelegte „temp“ Ordner am Ende nicht mehr gebraucht wird, wird dieser gelöscht. Der letzte Schritt fasst, falls die Checkbox „summarize all files after download“ markiert worden ist, die für jede DOI erstellte csv Datei mit allen Autoren zu einer kompletten Datei zusammen. Die in der GUI eingestellten Spalten werden in dieser Datei übernommen und jene die nicht gewählt wurden entfernt.

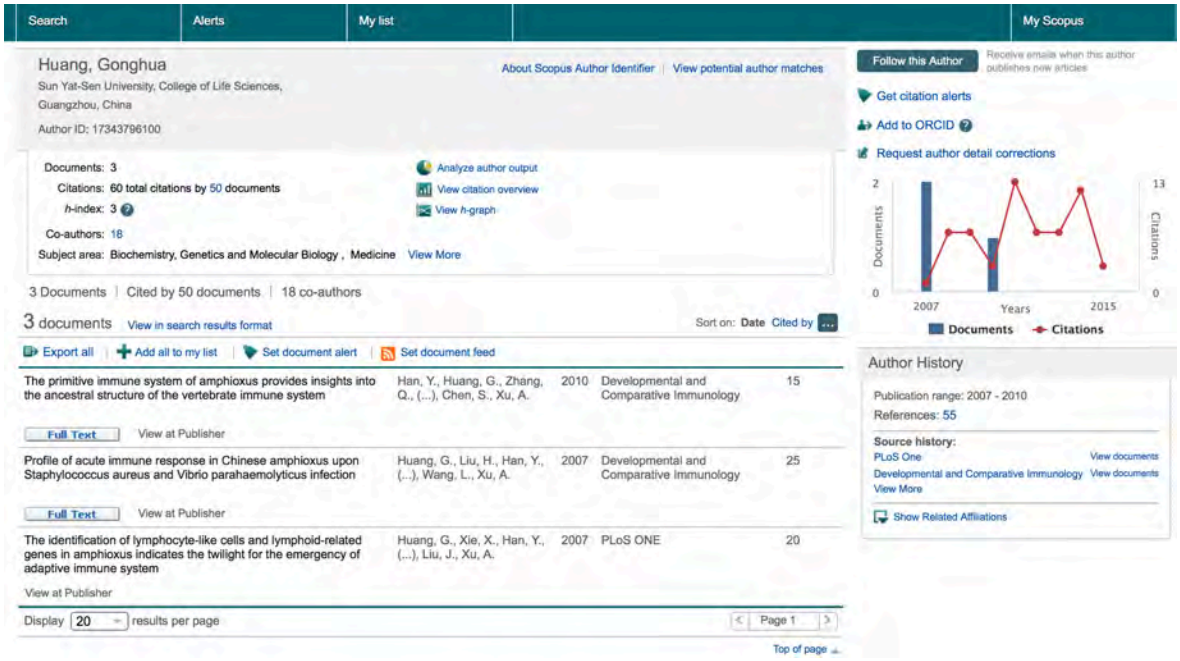

Abbildung 5: Scopus Webseite eines Autors[4]

Da die fertige Datei beliebig groß werden kann, wird diese je nach Spaltenanzahl in mehrere einzelne Dateien gesplittet. Nachdem das Erstellen und Zusammenfassen abgeschlossen ist, wird eine Datei „summary.csv“ im Arbeitsverzeichnis erstellt, welche dem Benutzer eine Zusammenfassung über den Crawlvorgang liefert. Dabei werden alle DOIs aufgelistet, ob sie erfolgreich gecrawlt wurden und ob sie vollständig sind. Während der Laufzeit wird der Benutzer über Fehler, Erfolge oder Informationen im Log Bereich informiert. Er kann den Vorgang durch klicken auf „abort“ abbrechen. Nach betätigen des Buttons crawlt das Programm die sich aktuell in Arbeit befindlichen DOIs fertig und stoppt anschließend. Während der Programmlaufzeit sollte jegliche Interaktion mit den sich öffnenden Firefox Fenster vermieden werden.

[4]Scopus Autor Huang, Gonghua, vgl. `http://www.scopus.com.scopeesprx.elsevier.com/authid/detail.url?authorId=17343796100&eid=2-s2.0-46449129564`, 06.02.2015

3.7 Verwendete Libraries

- **Osermiller CSV Parser 1.08.02**
  Der Ostermiller CSV Parser wurde zum Einlesen von csv Dateien beim Programmstart verwendet. Dabei werden die einzelnen DOIs durch aufsplitten der Datei nach jedem Semikolon gewonnen und in eine Liste gespeichert, damit das Programm die Daten weiter verarbeiten kann.

- **Java Excel API 2.6.12**
  Ähnlich wie der Ostermiller CSV Parser dient die Java Excel API dem Einlesen und Speichern der DOIs beim Programmstart. Die Einzelnen DOIs werden dabei auch in einer Liste gespeichert. Zu beachten ist dabei, dass sich die Tabelle auf der ersten Mappe befindet.

- **open CSV 3.1**
  Im Gegensatz zum Osermiller CSV Parser, welcher nicht mit Anführungszeichen in csv Dateien richtig umgehen kann, wird open CSV für das Einlesen und Bearbeiten der heruntergeladenen Dateien verwendet. Dieser escaped die Anführungszeichen in den einzelnen Zellen der csv Datei richtig und sorgt damit für die richtige Ausgabe in den bearbeiteten Dateien.

- **Selenium 2.44.0** Da Scopus keinen direkten Zugriff auf seine Datenbank mit SQL erlaubt, erfolgt der Datenzugriff durch Automatisierung des Browsers. Selenium interagiert mit Scopus als wäre es ein Benutzer und ist somit von der Webseite nur schwer als Programm erkennbar. Als Browser wurde Firefox für die Automation verwendet, da sich Einstellungen, wie zum Beispiel Speicherort der heruntergeladenen Dateien und Dateiendungen, die ohne weitere Interaktion heruntergeladen werden können, einstellen lassen.

3.8 Ordnerstruktur

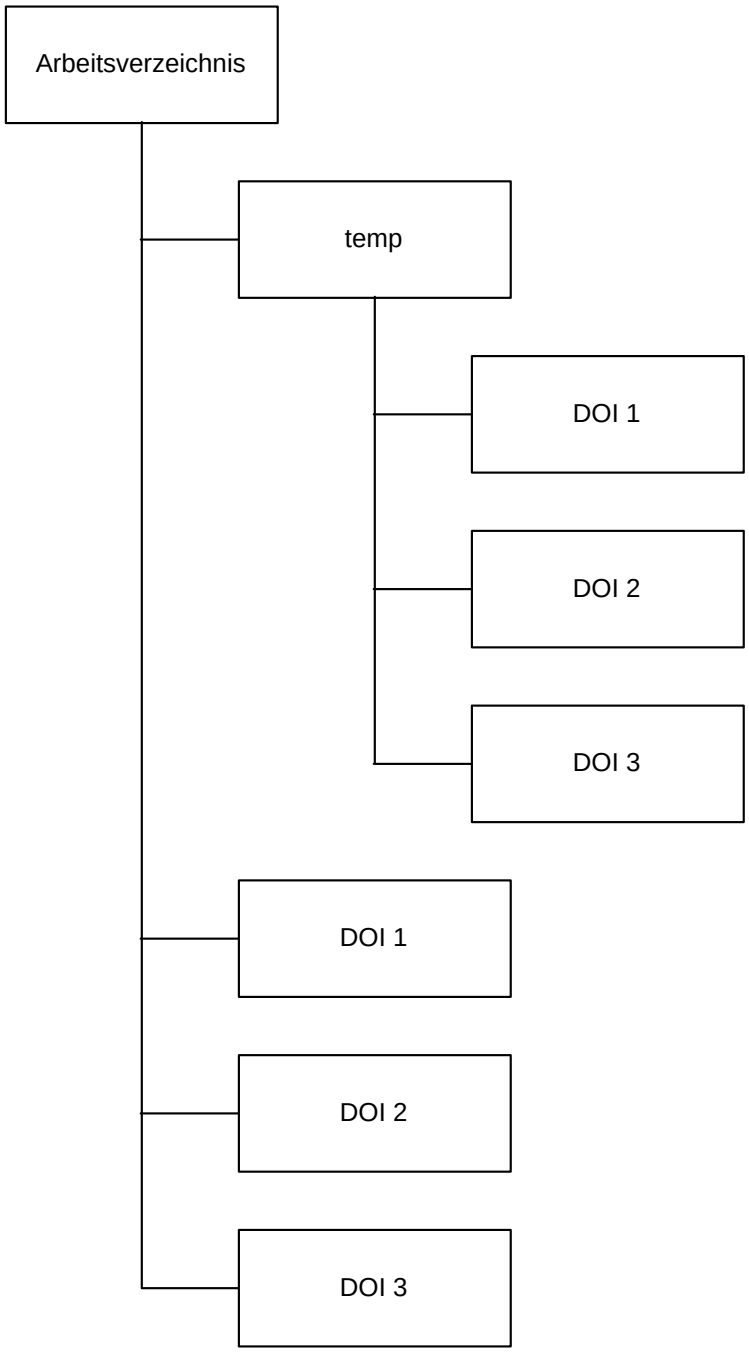

Abbildung 6: Die Ordnerstruktur, die das Programm beim Ausführen von selbst anlegt.

Die obere Abbildung zeigt die Visualisierung der Ordnerstruktur, welche von dem Crawler bei der Laufzeit angelegt wird. Das Arbeitsverzeichnis wird zu Beginn des Prozesses vom Programm, in dem vom Benutzer ausgewählten Ordner in der GUI, erstellt. Der Name dieses Ordners setzt sich aus der Zeichenkette „doiCrawler_“ und dem Namen der Datei, mit den DOIs beziehungsweise der einzelnen DOI zusammen. Der angelegte „temp“ Ordner beinhaltet für jede DOI einen weiteres Verzeichnis. In diesen Verzeichnissen werden die temporären csv Dateien, welche von Scopus für jeden

Autor heruntergeladen werden, gespeichert. Der Titel dieser temporären Ordner entspricht den Namen der DOIs. Im Arbeitsverzeichnis selbst werden ebenfalls Ordner für jede DOI erstellt. Diese beinhalten die fertig bearbeiteten csv Dateien. Dabei handelt es sich um eine csv pro Autor und eine komplette csv mit Informationen von allen Autoren. Die finale Datei, die die zusammengefassten Informationen von allen DOIs enthält, wird zum Schluss des Crawlvorgangs im Arbeitsverzeichnis abgespeichert. Eine Zusammenfassung des kompletten Vorgangs wird ebenfalls dort erstellt.

3.9 Ausgabedateien

Falls die Checkbox zum Aggregieren markiert worden ist, werden die einzelnen csv Dateien für jede DOI zusammengefasst. Die Ausgabedatei beziehungsweise die Ausgabedateien des Crawlers beinhalten sämtliche Informationen, über alle Publikationen der Autoren, der zuvor gewählten DOIs. Der Name der Datei/Dateien lautet „complete_NR_doiCrawler_DATEINMAE.csv“, wobei NR für die Nummerierung bei mehreren Dateien steht und DATEINAME für den Namen der Datei mit den DOIs beziehungsweise für die einzelne DOI. Der Benutzer kann in der GUI den Detailgrad der Ausgabedateien durch klicken auf „columns“ wählen. Dadurch werden nicht benötigte Informationen weggelassen und es lässt sich Speicherplatz einsparen. Je nach Anzahl von Spalten (Informationen) der Ausgabedatei, wird diese nach einer bestimmten Menge von Zeilen, aufgesplittet. Eine Zeile steht dabei für eine Publikation. Die nachfolgende Tabelle zeigt, wie sich die Spaltenanzahl zur Menge von Zeilen verhält.

Anzahl der Spalten	Zeilen nach denen aufgesplittet wird
2 - 9	50000
10 - 17	25000
18 - 29	10000
30 - 43	5000

Tabelle 1: Splittübersicht der finalen Datei

Dabei gilt es zu beachten, dass die von dem Programm hinzugefügten Spalten („crawler_doi“ und „crawler_position“) auch mitgezählt werden.
Die von Scopus heruntergeladenen csv Dateien besitzen 41 Spalten. In der folgenden Tabelle werden diese aufgelistet.

crawler_doi	authors with affiliations	conference date
crawler_position	abstract	conference location
authors	author keywords	conference code
title	index keywords	ISSN
year	molecular sequence numbers	ISBN
source title	chemicals/CAS	CODEN
volume	trademarks	DOI
issue	manufacturers	PubMed ID
art. no	funding details	language of original document
page start	references	abbreviated source title
page end	correspondence address	document type
page count	editors	source
cited by	sponsors	EID
link	publisher	-
affiliations	conference name	-

Tabelle 3: Übersicht aller wählbaren Spalten

Die fett gedruckten Spalten „crawler_doi“ und „crawler_position“ werden vom Programm zu den vorhanden Informationen in der csv Datei hinzugefügt. Die restlichen Splaten, die von Scopus selbst in die csv Datei eingetragen werden, werden nicht weiter eklärt, da diese selbsterklärend sind.
Die Spalte **crawler_doi** bezieht sich auf die DOI der Publikation von welcher der Autor gecrawlt worden ist.
Die Spalte **crawler_position** bezieht sich auf die Autorposition der Publikation.
Wenn zum Beispiel nach den Autorinformationen der Publikation mit der DOI „10.1371/journal.pone.0001301“ gesucht wird, wird für jede Publikation von allen Autoren dieser DOI, die gesuchte „10.1371/journal.pone.0001301“ in die Spalte eingetragen. Wenn der aktuelle Autor an der dritten Position steht, wird in die Spalte crawler_position eine drei eingetragen.
Eine weitere Datei, die vom Programm erstellt wird ist die „log_scheduler.txt“. Diese Logdatei beinhaltet sämtliche Informationen zu dem gesamten Crawlvorgang. Es werden Informations Meldungen, wie das erfolgreiche erstellen diverser Ordner, das erfolgreiche Einloggen in die Scopus Datenbank oder den erfolgreichen Download einer csv Datei mit dokumentiert. Fehler, die während der Laufzeit auftreten, werden in diesem File genauestens aufgelistet. Dazu gehört die genaue Angabe von Zeit, Klasse und Zeile, in der der Fehler aufgetreten ist und die geworfene Exception mit allen Informationen. Mit diesen Angaben lässt sich der Fehler genauestens nachverfolgen und vom Entwickler beheben.

Der letzte Schritt, den der Crawler vor dem Beenden des Vorgangs ausführt, ist das Erstellen der „summary.csv“ Datei. Dieses File liefert eine Übersicht über den kompletten Crawlvorgang und zeigt dem Anwender, ob es Fehler gab oder Autorinformationen nicht heruntergeladen werden konnten.

4 Probleme bei der Implementierung

Im folgenden Abschnitt werden Probleme näher erläutert, welche bei der Implementierung des Crawlers gelöst wurden.

4.1 Kein direkter Datenbankzugriff

Noch vor Beginn der Implementierung stellte sich die Frage, wie die Daten aus der Scopus Datenbank automatisiert heruntergeladen werden können. Da Scopus keine direkten SQL Anfragen akzeptiert, sondern nur als normale Webseite dargestellt werden kann, musste eine Möglichkeit zur Browser Automatisierung gefunden werden. Wie im Kapitel Scopus DOI Crawler bereits beschrieben, wird die Selenium Library für diesen Zweck verwendet. Diese lässt alle aktuellen Browser, wie Firefox, Chrome oder Opera, durch Java Code steuern.

4.2 Finale Datei hat zu viele Zeilen beziehungsweise ist zu groß

Eine csv Datei hat grundsätzlich keine Einschränkung bei der Anzahl von Zeilen beziehungsweise von der Dateigröße. Jedoch können Programme wie Microsoft Excel nur eine bestimmte Anzahl von Zeilen in einer Datei verarbeiten/anzeigen. Die Dateigröße könnte Probleme mit dem Betriebssystem hervorrufen, da eventuell die maximale Dateigröße beschränkt ist. Dadurch, dass bis zu 1000 DOIs in einem Vorgang gecrawlt werden können, wächst die finale Datei sehr schnell an und könnte die zuvor aufgezeigten Probleme verursachen. Dadurch splittet das Programm die finale Datei je nach Spaltenanzahl auf. Die genaue Übersicht, nach welcher Spaltenanzahl wie gesplittet wird, wurde im vorherigen Kapitel bereits erklärt.

4.3 Liste von Autoren konnte nicht richtig ausgelesen werden

Der Crawler verwendet Firefox als Browser mit dem er ständig kommuniziert. Damit die richtigen Web Elemente von der Webseite ausgelesen werden, wurde das Selenium Firefox Plugin verwendet. Dies erlaubt dem Benutzer Abläufe zu dokumentieren und unterschiedliche Identifikatoren für jedes Element auszuwählen. somit können die richtigen Elemente in den Programm Code geladen und verwendet werden. Bei den Autoren handelt es sich um eine Liste von Links. Jedes Element wird dabei im Programm Code in eine Liste von Web Elementen gespeichert. Dadurch, dass bei machen Autoren noch ein E-Mail Symbol, zum kontaktieren des Autors, in der Aufzählung enthalten ist, verursachten diese Symbole immer Fehler, da nicht die erwartete Seite geladen wurde. Diese wurden durch dir Überprüfung des Textes der Elemente herausgefiltert. Da das E-Mail Symbol keinen Text besitzt, lies sich die Überprüfung durch einfaches filtern nach nicht leeren Textinhalten durchführen.

4.4 Anführungszeichen wurde nicht richtig escaped

Eine csv Datei ist ein Textdokument, das als Tabelle interpretiert wird. Eine Zeile in dem Dokument steht dabei für eine Zeile in der Tabelle. Am Ende einer jeden Zeile befindet sich ein Umbruch. Die Trennung der einzelnen Spalten wird durch ein Semikolon umgesetzt. Damit die Spaltentrennzeichen auch innerhalb einer Zelle verwendet werden können, setzt man am Anfang und Ende des gewünschten Zelleninhalts Anführungszeichen. Das Einlesen der heruntergeladenen csv Dateien mit dem Ostermiller csv Parser war aufgrund falschem escapen der Anführungszeichen immer fehlerhaft. Dadurch wurden die Zellen mit Anführungszeichen als Inhalt in mehrere Unterteilt und somit alle anderen Zellen verschoben. Der Fehler wurde durch eine andere Library, die die Anführungszeichen richtig escaped, behoben. Die openCSV Library hat keine Probleme mit Anführungszeichen in den Zellen und macht somit alles richtig.

4.5 csv Dateien wurden nicht richtig vollständig eingelesen

Nachdem eine csv Datei mit Autorinformationen heruntergeladen wurde, fügt das Programm die zuvor erwähnten Spalten „crawler_doi“ und „crawler_position“ am Anfang der Tabelle hinzu. Dies kann nur durch vorheriges Importieren der Datei erfolgen. Die opencsv Library kann große Dateien nicht vollständig einlesen. Dadurch, dass das Programm über jede Zeile iteriert und diese einlest, lässt sich der Verlust der Informationen verhindern. Somit wird der gesamte Inhalt richtig in das Programm geladen. Dieser Fehler wurde unter anderem auch in weiteren Methoden der Export Klasse behoben.

4.6 Crawlvorgang beschleunigen

Nachdem die Login Daten und das DOI File überprüft wurde, startet der eigentliche Vorgang. Dabei werden bei jeder DOI die Informationen zu allen Autoren von Scopus heruntergeladen. Um den Crawlvorgang zu beschleunigen, werden bis zu drei Publikationen gleichzeitig bearbeitet. Dies erfolgt durch eine „scheduler“ Klasse, die alle Vorgänge überwacht und den gesamten Ablauf regelt. Dadurch, dass die Parallelisierung bis zu drei gleichzeitige Vorgänge erlaubt, verkürzt sich die Zeit um knapp zwei drittel.
Durch das klicken auf den Namen eines Autors bei einer Publikation gelangt man zur Übersichtsseite des Autors. Dort werden alle seine Veröffentlichungen aufgelistet. Dadurch, dass der Crawler die Exoprt Funktion von Scopus nutzt, wird erhebliche Zeit eingespart. Ohne diese Funktion müsste der Crawler sämtliche Detailseiten der Publikationen durchklicken und die Informationen extrahieren.

4.7 Autorinformationen von mehr als 2000 Einträgen können nicht heruntergeladen weren

Eine Veröffentlichung hat oft mehrere Autoren. Diese können beliebig viele Publikationen veröffentlicht haben. Der Crawler verwendet die Export Funktion von Scopus um, wie zuvor beschrieben, die Abarbeitungszeit der Vorgänge zu beschleunigen. Da diese Funktion nur für eine Publikationsanzahl von bis zu 2000 zur Verfügung steht, kann der Crawler nicht alle Informationen von Autoren mit mehr als 2000 Publikationen verarbeiten. Der Benutzer wird im Log Bereich der GUI beziehungsweise im summary File darüber informiert. Er kann die fehlenden Informationen manuell bei Scopus anfordern.

4.8 Webseite liefert Ressourcen zu langsam

Durch das Parallelisieren des Crawlvorgangs wird der Internet Traffic der lokalen Anbindung erhöht. Dies führt dazu, dass die Ressourcen von Scopus langsamer geladen werden. Um einen Timeout zu vermeiden, wurden die Wartezeiten auf die Webelemente von Anfangs zehn auf 60 Sekunden erhöht. Dadurch lassen sich die meisten Fehler dieser Art vermeiden.

4.9 Fehler während des Vorgangs

Trotz diverser Schutzvorkehrungen, wie zum Beispiel erhöhen des Timeouts, lassen sich oft diverse Fehler nicht vermeiden. Die häufigste Form von Problemen ist das nicht auffinden von Web Elementen auf der Scopus Webseite. Diese Elemente werden innerhalb des Timeout Zeitraums nicht geladen und verursachen somit Schwierigkeiten. Wenn der Crawlvorgang fehlerhaft ist, wird dieser abgebrochen. Das Programm startet anschließend bis zu zwei weitere Versuche, um die gewünschten Informationen zu finden.

4.10 Bei einem Fehler konnte das Programm nicht erneut gestartet werden

Damit das Programm kein zweites mal parallel gestartet werden kann, wird zu Beginn ein Port geöffnet. Der Crawler überprüft ob der Port geöffnet ist und schließt das Programm falls dies der Fall ist. Bei einem Absturz kann es sein, dass ein Firefox Thread im Hintergrund weiter läuft und es den Anschein hat, dass Programm sei beendet, was allerdings nicht der Fall ist. Der Port bleibt geöffnet und der Crawler kann nicht erneut gestartet werden. Aus diesem Grund wurde ein Shutdown Hook implementiert, welcher alle laufenden Firefox Threads beendet und den beim Starten geöffneten Port schließt. Somit kann das Programm auch bei einem Absturz wieder neu gestartet werden.

4.11 Der Speicher der JVM reicht nicht aus

Wenn ein Vorgang mit sehr vielen DOIs (500 oder mehr) durchgeführt wird, kann das zu einem Absturz des Programms führen. Dies liegt an dem zugeteilten Speicher der JVM, welcher beim Einlesen der einzelnen, zusammengefassten DOI Dateien zu wenig wird. Dieses Problem lässt sich nur schwer beheben ohne die JVM zu verändern. Um dennoch den Fehler zu vermeiden, wurde die maximale Anzahl an DOIs auf 200 minimiert. Falls dennoch ein Speicherfehler auftritt, wird dieser in der Log Datei und im Log Bereich der GUI angezeigt. Der Programmabsturz wird durch das fangen des Fehlers und dem geregelten beenden des Programms verhindert. Falls der Benutzer auf das Aggregieren der Dateien verzichten möchte, kann er dies durch deaktivieren der Checkbox in der GUI vor dem Starten des Vorgangs. Danach lassen sich Dateien mit bis zu 5000 DOIs auswählen.

4.12 Firefox wird bei einem Fehler nicht beendet

Um eine große Anzahl an geöffneten Firefox Fenstern/Prozesse zu vermeiden, wird nach jeder gecrawlten DOI der jeweils nicht mehr benötigte Thread beendet. Falls ein Fehler auftritt, wird der ursprüngliche Programmcode nicht mehr ausgeführt und der Prozess wird möglicherweise nicht beendet. Um diesem Problem entgegenzuwirken wird in jedem „catch“ Block der Firefox Prozess, in dem der Fehler auftrat, geschlossen. Des weiteren werden am Ende des Crawlvorgangs, beziehungsweise beim Schließen des Programms, alle gestarteten Firefox Threads durchgegangen und falls noch am Leben, beendet.

4.13 Seite mit den Autor-/Publikationsinformationen wird nicht vollständig geladen

Damit der Crawler alle Publikationen eines Autors herunterladen kann, muss zuerst auf die Publikations- und anschließend auf die Scopus Webseite des gewünschten Autors navigiert werden. Wenn sehr viele DOIs hintereinander gecrawlt werden sollen, kann es zum nicht vollständigen Laden der einzelnen Seite kommen. Dadurch lassen sich Web Elemente, wie die Liste von Autoren beziehungsweise der Link zum Herunterladen der Autorinformationen welcher den Datei Download zur Verfügung stellt, nicht lokalisieren und der Crawler bricht den Vorgang ab. Um diesem Fehler vorzubeugen wird ein Reload der Webseite durchgeführt.

5 Beispiel

In diesem Kapitel werden die Ergebnisse eines Crawlvorgangs zum besseren Verständnis der Funktionsweise und des Aufbaus dargestellt.

5.1 Eingaben

Damit der Crawler arbeiten kann, benötigt er gültige Login Daten für die Scopus Webseite (`http://scopees.elsevier.com/`). Als Input wurde eine csv Datei (only3.csv) mit drei DOIs (10.1371/journal.pone.0000202, 10.1371/journal.pone.0000203, 10.1371/journal.pone.0000206) verwendet. Dabei wurden die Spalten „authors“, „title“ und „year“ ausgewählt. Die beiden Spalten „crawler_doi“ und „crawler_position“ werden vom Programm automatisch hinzugefügt und können in der Splatenauswahl nicht deaktiviert werden.

5.2 GUI Log Bereich

Log-Area

```
09:58:15 app started
----------
09:59:45 checking login data
09:59:45 folder "doiCrawler_only3csv" created
09:59:55 login data correct
09:59:55 reading file
09:59:55 read file successfully
09:59:55 start crawling
10:00:10 error while crawling 10.1371/journal.pone.0000203
10:00:10 try to crawl 10.1371/journal.pone.0000203 again
10:00:16 error while crawling 10.1371/journal.pone.0000203
10:00:16 try to crawl 10.1371/journal.pone.0000203 again
10:00:24 error while crawling 10.1371/journal.pone.0000203
10:01:53 success crawling 10.1371/journal.pone.0000202
10:07:20 file not downloaded. May more than 2000 entries 10.1371/journal.pone.0000206
         position 8
10:12:41 success crawling 10.1371/journal.pone.0000206
10:12:41 try to write complete files
10:13:12 done
>>>>>>>>>>
error 10.1371/journal.pone.0000203 -
success missing 10.1371/journal.pone.0000206 information for author position [8] missing
success 10.1371/journal.pone.0000202 all downloaded
>>>>>>>>>>
```

Der oben stehende Text, wurde während des Crawlvorgangs in dem Log Bereich der GUI ausgegeben. Dabei ist auffallend, dass mehrere Fehler beim Crawlen der DOI 10.1371/journal.pone.0000203 aufgetreten sind. Nach dem dritten Fehlversuch bricht das Programm ordnungsgemäß ab und überspringt die DOI. Die letzten fünf Zeilen fassen den gesamten Vorgang zusammen und zeigen dem Benutzer welche DOIs erfolgreich geladen wurden und bei welchen Autorinformationen fehlen.

5.3 Ausgabe: log_scheduler.txt

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE log SYSTEM "logger.dtd">
<log>
<record>
  <date>2015-03-18T09:59:45</date>
  <millis>1426669185633</millis>
  <sequence>0</sequence>
  <logger>DOI Crawler</logger>
  <level>INFO</level>
  <class>logic.Scheduler</class>
  <method>checkLogin</method>
  <thread>45</thread>
  <message>try logging in</message>
</record>
<...>
<record>
  <date>2015-03-18T10:00:10</date>
  <millis>1426669210903</millis>
  <sequence>33</sequence>
  <logger>DOI Crawler</logger>
  <level>SEVERE</level>
  <class>logic.Crawler</class>
  <method>crawlDoi</method>
  <thread>55</thread>
  <message>firefox error 10.1371/journal.pone.0000203 Error communicating
  with the remote browser. It may have died.
  Build info: version: '2.44.0', revision: '76d78cf',
  time: '2014-10-23 20:03:00'
  System info: host: 'Manuels-MBP.fritz.box', ip: '192.168.178.34',
  os.name: 'Mac OS X', os.arch: 'x86_64', os.version: '10.10.2',
  java.version: '1.8.0_31'
  Driver info: driver.version: RemoteWebDriver</message>
  <exception>
    <message>org.openqa.selenium.remote.UnreachableBrowserException:
    Error communicating with the remote browser. It may have died.
    Build info: version: '2.44.0', revision: '76d78cf',
    time: '2014-10-23 20:03:00'
```

```
38      System info: host: 'Manuels-MBP.fritz.box', ip: '192.168.178.34',
39      os.name: 'Mac OS X', os.arch: 'x86_64', os.version: '10.10.2',
40      java.version: '1.8.0_31'
41      Driver info: driver.version: RemoteWebDriver</message>
42      <frame>
43        <class>org.openqa.selenium.remote.RemoteWebDriver</class>
44        <method>execute</method>
45        <line>593</line>
46      </frame>
47      <frame>
48        <class>org.openqa.selenium.remote.RemoteWebDriver</class>
49        <method>findElement</method>
50        <line>352</line>
51      </frame>
52      <frame>
53        <class>org.openqa.selenium.remote.RemoteWebDriver</class>
54        <method>findElementByCssSelector</method>
55        <line>441</line>
56      </frame>
57      <frame>
58        <class>org.openqa.selenium.By$ByCssSelector</class>
59        <method>findElement</method>
60        <line>426</line>
61      </frame>
62      <frame>
63        <class>org.openqa.selenium.remote.RemoteWebDriver</class>
64        <method>findElement</method>
65        <line>344</line>
66      </frame>
67      <frame>
68        <class>logic.Crawler</class>
69        <method>searchForDoi</method>
70        <line>405</line>
71      </frame>
72      <frame>
73        <class>logic.Crawler</class>
74        <method>crawlDoi</method>
75        <line>136</line>
76      </frame>
77      <frame>
78        <class>logic.Crawler</class>
79        <method>run</method>
80        <line>48</line>
81      </frame>
82    </exception>
83  </record>
84  <...>
85  </log>
```

Listing 1: Auszug aus einer Log Datei, die während eines Vorgangs angelegt wird.

Damit der Entwickler aufgetretene Fehler im Programmablauf lokalisieren und im besten Fall bis zur nächsten Version beheben kann, schreibt das Programm bei jedem Vorgang ein Log Datei. Diese wird mit dem *java.util.logging.Logger* im Arbeitsverzeichnis erstellt.
Der oben stehende Auszug dieser Datei zeigt in der ersten Aufzeichnung (record) (Zeile 4 bis 14), den Versuch des Programms sich bei Scopus anzumelden. Dabei wird unter anderem das genaue Datum mit Zeitangabe, die Sequenz, die Klasse von der der Log Befehl kam, die Methode und die Nachricht mit dokumentiert. Bei dem Ersten record handelt es sich um einen simplen Info Eintrag.
Wie Fehler dokumentiert werden, zeigt der Zweite record (Zeile 16 bis 83). Dieser besitzt die gleichen Felder wie eine Info Aufzeichnung. Die Nachricht beinhaltet eine sehr genaue Beschreibung über den aufgetretenen Fehler. Dabei werden Variablen wie IP Adresse, Betriebssystem, Java Version aufgelistet. Die exakte Exception, welche man von der Konsole der Entwicklungsumgebung kennt, befindet sich ebenfalls in der Nachricht. In dieser werden alle aufgerufenen Methoden in den dazugehörigen Klassen unter genauer Zeilenangabe aufgelistet (Zeile 33 bis 82). Der Beispiel Fehler wurde bewusst durch das Beenden eines laufenden Firefox Threads hervorgerufen. Einer der häufigsten Fehler ist allerdings das nicht auffinden von bestimmten Web Elementen. Auf diesem Fehler wird im Kapitel „Probleme bei der Implementierung“ genauer eingegangen.

5.4 Ausgabe: summary.csv

status	doi	missing author information at position
error	10.1371/journal.pone.0000203	-
success missing	10.1371/journal.pone.0000206	[8]
success	10.1371/journal.pone.0000202	all author information downloaded

Tabelle 4: Zusammenfassung des gesamten Vorgangs

In der Datei „summary.csv“ lässt sich das Ergebnis des Crawlvorgangs nochmal nachlesen. Die Zusammenfassung bietet drei Informationen über jede einzelne DOI. Der Status zeigt dem Benutzer, ob die Dateien erfolgreich heruntergeladen werden konnten. Die zweite Spalte zeigt die jeweilige DOI an. Als letztes wird dem Anwender angezeigt, ob sämtliche Informationen von allen Autoren der jeweiligen DOI herunter geladen werden konnte oder nicht. Dies ist der Fall, wenn ein Autor mehr als 2000 Publikationen verfasst hat. Falls Autorinformationen fehlen erscheinen die Positionen

in dieser Spalte. Bei diesem Beispiel wurden die Informationen zur DOI 10.1371/journal.pone.0000203 aufgrund von Fehlern nicht heruntergeladen. Die Publikation mit der DOI 10.1371/journal.pone.0000206 wurde gecrawlt allerdings fehlen die Autorinformationen von Position 8. Die DOI in der letzten Zeile wurde komplett und richtig durchlaufen.

5.5 Ausgabe: complete_0_doiCrawler_only2csv.csv

crawler_doi	crawler_position	Authors	Title	Year
10.1371/journal.pone.0000202	1	Mols C.M.M., (...)	Great tits (Parus major) reduce (...)	2007
10.1371/journal.pone.0000202	1	Mols C.M.M., (...)	Assessing the reduction of (...)	2005
(...)	(...)	(...)	(...)	(...)
10.1371/journal.pone.0000202	2	Gienapp P., (...)	Why climate change will (...)	2014
10.1371/journal.pone.0000202	2	Reparaz L.B., Van Oers K., (...)	Mate preference of female (...)	2014
(...)	(...)	(...)	(...)	(...)
10.1371/journal.pone.0000206	1	Han Y., Huang G., (...)	The primitive immune system (...)	2010
10.1371/journal.pone.0000206	1	Huang G., Liu H., (...)	Profile of acute immune response (...)	2007
(...)	(...)	(...)	(...)	(...)
10.1371/journal.pone.0000206	7	Mizushima T., (...)	Phenotypes of dnaA mutants (...)	1996
10.1371/journal.pone.0000206	7	Shinpuku T., (...)	Phenotypes of dnaA mutants (...)	1995
(...)	(...)	(...)	(...)	(...)
10.1371/journal.pone.0000206	9	Shi Y., Li K., (...)	Three-dimensional visualization (...)	2015
10.1371/journal.pone.0000206	9	Guo F., Liu Z., (...)	Capsid expansion mechanism of (...)	2014
(...)	(...)	(...)	(...)	(...)
10.1371/journal.pone.0000206	13	Xu A., McKenna K., (...)	Sequencing and genetic analysis (...)	1993
10.1371/journal.pone.0000206	13	Xu A., Clark T.J., (...)	Sequencing and genetic analysis (...)	1991

Tabelle 5: Finale Datei eines Vorgangs

Die vorherige Tabelle im Querformat visualisiert einen Auszug aus einer aggregierten finalen Datei. Diese enthält sämtliche Informationen von allen DOIs. Die zuvor Ausgewählten und die vom Crawler hinzugefügten Spalten bilden die erste Zeile der csv Datei. Die Autorinformationen der DOI 10.1371/journal.pone.0000202 befinden sich in den Zeilen zwei bis sechs. Nachdem alle Autoren der ersten DOI aufgelistet wurden, folgen die Informationen der zweiten DOI 10.1371/journal.pone.0000206. Da bei dieser DOI der Autor an der Position 8 mehr als 2000 Publikationen veröffentlicht hat, konnte die csv Datei nicht heruntergeladen werden. Somit fehlen die Einträge zu diesem Autor. Das Dokument endet mit der letzten Publikation des letzten Autors der zweiten DOI. Da die Informationen zur dritten DOI nach dem dritten Versuch nicht geladen werden konnten, werden diese in der finalen Datei nicht aufgelistet.

6 Ausblick

Der Scopus DOI Crawler ist durch seinen strukturierten Aufbau auf eine sehr leichte Weise erweiterbar. Dadurch könnte das Programm auf andere Publikationswebseiten wie `http://www.sciencedirect.com` erweitert werden. Da diese Webseiten vermutlich über keine eigene Exportfunktion verfügen, müssten die gewünschten Informationen von der jeweiligen Webseite ausgelesen werden.
Eine zusätzliche Funktion zur automatischen E-Mail Benachrichtigung könnte ebenfalls implementiert werden. Diese würde nach Fertigstellung eines Vorgangs den Benutzer per Mail informieren. Aufgrund der langen Dauer eines Vorgangs mit vielen DOIs würde dies ständiges überprüfen der GUI überflüssig machen.
Um erneutes Eingeben der Login Daten beziehungsweise Auswählen der gewünschten Spalten bei jedem Start des Programms zu vermeiden, könnte eine properties Datei die gewünschten Werte speichern. Diese Funktion wäre in Kombination mit der E-Mail Benachrichtigung sehr sinnvoll, da die genauen E-Mail Server Einstellungen nur einmal vom Benutzer eingegeben werden müssten.
Eine weitere denkbare Evolution des Programms wäre das Speichern der Daten in einer Datenbank. Dadurch würden sich einige Fehler und Probleme, wie die Speicherknappheit oder die zu großen aggregierten Dateien, vermeiden. Der Zugriff könnte über ein eigenes Programm oder auch über eine Webseite erfolgen.

7 Fazit

Trotz vieler Probleme während der Implementierung entwickelte sich der DOI Crawler zu einem zuverlässigen Programm, welches für das Sammeln von Autorinformationen in der Scopus Datenbank eingesetzt weren kann. Dadurch, dass die Weiterentwicklung aufgrund der Struktur des Programms ebenfalls gegeben ist, kann dieser auf weitere Webseiten angepasst werden. Dies macht den Crawler zu einem universell einsetzbaren Informationsbeschaffer im Internet.

Literaturverzeichnis

[MK00] Mei Kobayashi, Koichi T.: Information Retrieval on the Web. In: *ACM Computing Surveys* 32 (2000), S. 145 – 173

[SD11] S.S. Dhenakaran, K. Thirugnana S.: WEB CRAWLER - AN OVERVIEW. In: *International Journal of Computer Science and Communication* 2 (2011), S. 265 – 267

[TVU14] Trupti V. Udapure, Rajesh C. D. Ravindra D. Kale K. Ravindra D. Kale: Study of Web Crawler and its Different Types. In: *IOSR Journal of Computer Engineering* 16 (2014), S. 1 – 5

Anhang

Scopus Webseite
`http://scopees.elsevier.com`, Zugriff: 19.01.2015

Links zur verwendeter Software

- **Ostermiller CSV Parser**
  Webseite: `http://ostermiller.org/utils/CSV.html`, Zugriff 05.01.2015
- **Java Excel API**
  Webseite: `http://jexcelapi.sourceforge.net`, Zugriff 05.01.2015
- **open CSV**
  Webseite: `http://opencsv.sourceforge.net`, Zugriff 05.01.2015
- **Selenium**
  Webseite: `http://www.seleniumhq.org`, Zugriff: 05.01.2015
- **Java Runtime Environment**
  Webseite: `https://www.java.com/de/download/`, Zugriff 05.01.2015
- **Java Development Kit 8**
  Webseite: `http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html`, Zugriff 05.01.2015
- **Eclipse**
  Webseite: `https://eclipse.org`, Zugriff 05.01.2015

Eidesstattliche Erklärung

Eidesstattliche Erklärung zur Bachelorarbeit

Ich versichere, die von mir vorgelegte Arbeit selbstständig verfasst zu haben. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder nicht veröffentlichten Arbeiten anderer entnommen sind, habe ich als entnommen kenntlich gemacht. Sämtliche Quellen und Hilfsmittel, die ich für die Arbeit benutzt habe, sind angegeben. Die Arbeit hat mit gleichem Inhalt bzw. in wesentlichen Teilen noch keiner anderen Prüfungsbehörde vorgelegen.

$Unterschrift$: $Ort, Datum$: