Technische Universität Darmstadt
Fachbereich Informatik
Fachgebiet Knowledge Engineering

Zitationsanalyse für das Gebiet „Lernen in Spielen“

Diplomarbeit von Michael Stegbauer

Betreut von Prof. Dr. Johannes Fürnkranz

Kurzzusammenfassung

Das Fachgebiet Knowledge Engineering pflegt eine umfangreiche Literatursammlung zum Thema „Maschinelles Lernen in Spielen“. Ziel dieser Diplomarbeit ist es auf Basis dieser Daten sowie mit Daten aus externen Zitationsdatenbanken weitere Informationen zu diesem Themengebiet zu ermitteln. Zu diesem Zweck werden in dieser Arbeit allgemeine Methoden zum Datenabgleich mit unterschiedlichen bibliographischen Datenbanken sowie verschiedene graphenbasierte Analyseverfahren, wie PageRank, HITS und PHITS beschrieben und in Form der Anwendung Citana implementiert. Dieses Programm wird abschließend für die speziellen Untersuchungen an der Literatursammlung genutzt und die Ergebnisse vorgestellt.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1. Einleitung

Wenn ich weiter als andere gesehen habe, dann nur deshalb,
weil ich auf der Schulter von Giganten stand.

Sir Isaac Newton (1643 - 1727)

1.1. Motivation

Wissenschaftliche Forschung ist nur durch die Verwendung bereits vorhandener Erkenntnissen effektiv durchführbar. Dementsprechend wichtig ist die Veröffentlichung von Forschungsergebnissen für die Wissenschaft. Ebenso wichtig ist die korrekte Zitierung von fremden Aussagen um zum einen die eigene Arbeit nachprüfbar zu machen und zum anderen die fremde Arbeit anzuerkennen.

Da die Anzahl von wissenschaftlichen Publikationen stetig wächst und die Themengebiete einem ständigen Wandel unterlegen sind, ist die Orientierung bei der Suche nach passender Literatur zu einem Thema oft schwierig. Aus diesem Grund existieren Zitationsdatenbanken, wie zum Beispiel *CiteSeer* [1], welche eine große Anzahl an Dokumenten sowie ihren Zitierungen untereinander erfassen. Die Einsatzgebiete solcher Datenbanken können in die Bereiche *Information Retrieval* und *Data-Mining* eingeordnet werden. Für das Information Retrieval werden relevante Dokumenten für eine bestimmte Suchanfrage gesucht. Im Data-Mining wird dagegen versucht Muster in den Daten zu erkennen, also bspw. einzelne Themengebiete oder Forschergruppen zu finden.

CiteSeer stellt in erster Linie eine Suchmaschine für wissenschaftliche Dokumente und damit eine Anwendung des Information Retrievals dar. Die Bewertung der Suchergebnisse basiert auf der Anzahl der Zitierungen einer Publikation und zeigt nur eine allgemeine Bedeutung in der Datenbank an. Allgemein fehlen Funktionen in CiteSeer um mit einer Menge von Einträgen, welche beispielsweise ein Themengebiet beschreiben können, arbeiten zu können. So ist es zwar möglich ähnliche Einträge für ein bestimmtes Dokument zu suchen, aber es ist nicht vorgesehen eine Sammlung von Publikationen für eine solche „Suchanfrage“ zu benutzen. Funktionen zur Erkennung von Mustern in einer solchen Dokumentenmenge fehlen ebenfalls. Ähnliche Einschränkungen gelten für viele der verfügbaren Zitationsdatenbank.

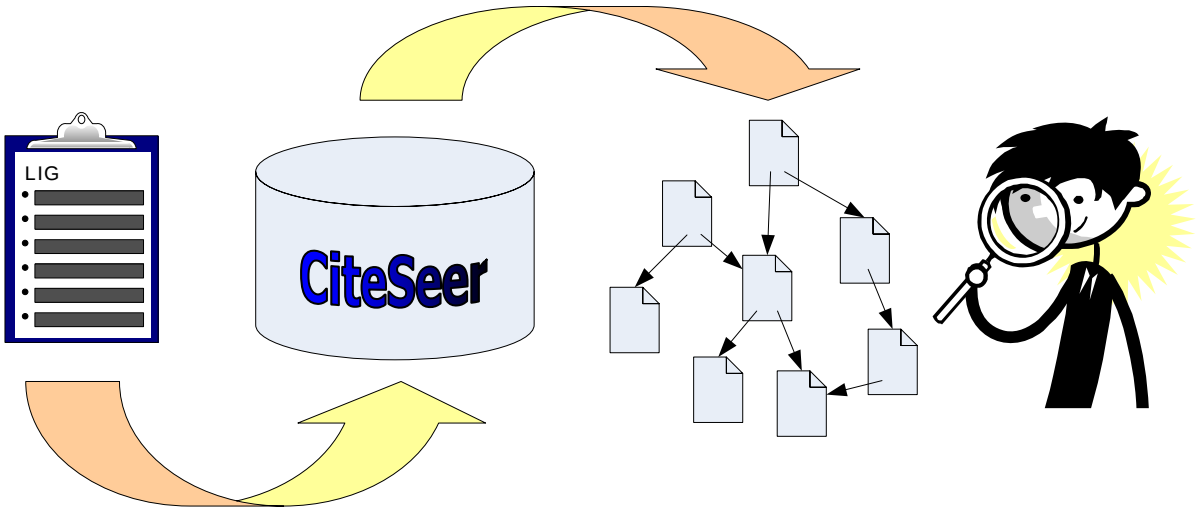


Abbildung 1.1.: Exemplarischer Arbeitsablauf für die Analyse einer Literatursammlung

1.2. Ziele der Arbeit

Das Fachgebiet Knowledge Engineering pflegt eine umfangreiche Literatursammlung zum Thema „Maschinelles Lernen in Spielen“. Aufgabe dieser Diplomarbeit ist es, für diese Literatursammlung eine Analyse der Referenzen zu erstellen und daraus Erkenntnisse zu gewinnen, welches die wichtigsten Untergebiete dieses Feldes sowie die wichtigsten Publikationen für jedes dieser Untergebiete sind. Zusätzlich sollen möglichst noch wichtige Publikationen gefunden werden, die noch nicht in der Datenbank enthalten sind.

Die dazu nötigen Verarbeitungsschritte (vgl. Abbildung 1.1) bestehen aus dem Finden der Publikationen und deren Referenz-Information in einer Zitationsdatenbank wie CiteSeer, dem Aufbau eines Zitationsgraphen und dem anschließenden Untersuchen desselbigen mit Methoden aus dem Data-Mining oder auch des Web-Mining, wie beispielsweise Hubs und Authorities.

1.3. Aufbau und Gliederung

Einen Überblick der benutzten Begriffe und derzeitigen bibliometrischen Anwendungen in Form von Zitationsdatenbanken wird in Kapitel 2 gegeben.

In Kapitel 3 werden die einzelnen benötigten Arbeitsschritte für die Zitationsanalyse theoretisch erläutert. Diese beginnen mit den Methoden zum Abgleichen von verschiedenen Datenquellen im Abschnitt „Duplikaterkennung“. Die folgenden beiden Abschnitte stellen Zitationsgraphen und den in dieser Arbeit benutzte thematische Zitationsgraphen mit seinen Analysemöglichkeiten genauer vor. Die folgenden Abschnitte beschäftigen sich dann konkret mit den verschiedenartigen Analysemöglichkeiten. Dies sind zum

einen sammlungsbasierte Bewertungen und zum anderen graphenbasierte Bewertungen, wie die aus dem Web-Mining bekannten Verfahren PageRank und HITS. Im Speziellen wird abschließend das PHITS-Verfahren erläutert, welches eine Möglichkeit bietet auch evtl. vorhandene Unterthemengebiete zu erkennen.

In Kapitel 4 wird das im Rahmen dieser Arbeit erstellte Programm *Citana* vorgestellt. Dazu wird eine kurze Einführung in die Benutzung gegeben, welches ebenfalls einen Einblick der verschiedenen Fähigkeiten darstellt. Die folgenden Abschnitte geben einen Überblick der Implementierung selbst. Vorgestellt werden hierbei nur die wichtigsten Datenstrukturen und der grundlegende Aufbau der graphischen Benutzeroberfläche.

Kapitel 5 beschreibt einige mit Citana durchgeführte Versuche und deren Ergebnisse. Unter anderem wird dort auch die Literatursammlung zum Thema „Maschinelles Lernen in Spielen“ untersucht.

Abschließend werden im Kapitel 6 die Ergebnisse dieser Arbeit zusammengefasst und ein Ausblick auf mögliche weiterführende Arbeiten gegeben.

2. Begriffe und verwandte Arbeiten

2.1. Bibliographien, Dokumente, Referenzen und Zitationen

Unter einer Bibliographie versteht man eine Zusammenfassung von Literaturangaben zu einem Thema oder unter einem bestimmten Aspekt (z. B. Literatur eines bestimmten Autors). Dabei wird keine Rücksicht darauf genommen, wo sich die zugehörigen Dokumente konkret befinden oder verfügbar sind, wodurch sie sich bspw. von einem Bibliothekskatalog unterscheidet. Die einzelnen Publikationen werden gewöhnlich durch eine Menge von Metadaten eindeutig beschrieben. Zusätzlich können auch weitere Daten, wie Bewertungen, Kommentare oder Stichwörter verzeichnet sein.

Gegenstand der Untersuchung in dieser Arbeit sind ausschließlich Bibliographien von wissenschaftlichen Publikationen. Eine wissenschaftliche Publikation werden wir hier oft einfach als Dokument oder auch nur als Publikation bezeichnen. Eine solche wissenschaftliche Arbeit zeichnet sich durch genaue Angabe der benutzten Quellen aus, was die Grundlage der Zitationsanalyse ist. Zu unterscheiden sind die Begriffe Referenzen und Zitationen eines Dokumentes. Eine Referenz ist ein Verweis auf eine im Text zitierte Arbeit. Im Gegensatz dazu stellt eine Zitation eine Referenz einer anderen Arbeit auf das Dokument dar. Beide Begriffe bezeichnen also eine „Zitiert“-Beziehung mit lediglich unterschiedlicher Sichtweise (vgl. Abbildung 2.1). Eine Referenz ist dabei eine aktive Eigenschaft, da diese vom Autor bestimmt wird und eine Zitation eine passive Eigenschaft, welche in der Regel vom Autor nicht direkt beeinflußt werden kann. Bis auf wenige Ausnahmen sind Referenzbeziehungen zeitlich gerichtet, da normalerweise nur bereits veröffentlichte Arbeiten zitiert werden. In einem gewissen Rahmen werden aber auch schon als Vorabdrucke vorhandene Dokumente zitiert, sodass hier theoretisch auch zukünftigen Arbeiten als Referenzen vorkommen können.

2.2. Bibliometrie und Zitationsanalyse

Die Bibliometrie befasst sich mit der quantitativen Bewertung von (wissenschaftlichen) Publikationen mittels statistischer Verfahren. Einige typische Fragestellungen betreffen die maßgebenden Veröffentlichungen, Zeitschriften, Forschungsinstitute oder Autoren in einem Fachgebiet. In der wissenschaftlichen Literatur werden traditionell Zitationen als eine Bewertungsgrundlage herangezogen, worauf schließlich die Disziplin der Zitationsanalyse aufbaut.

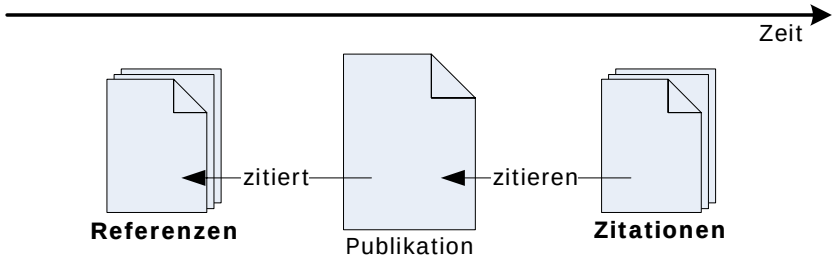


Abbildung 2.1.: Beziehungen zwischen einer Publikation, ihren Referenzen und ihren Zitationen

Als elementare Grundlage wird hierbei eine Zitierung als ein gegenseitiger Austausch zwischen zwei Dokumenten angesehen. Das zitierende Dokument bezieht von seiner Referenz Informationen und gibt ihr dafür eine öffentliche Anerkennung, worauf sich die Reputation der Referenz aufbaut (vgl. Abbildung 2.2).

Die Zitationsanalyse untersucht diese Art von wissenschaftlichem Kommunikationsprozess im Hinblick auf verschiedene Fragestellungen. Traditionell wird sie dazu benutzt um wissenschaftliche Leistung oder Bedeutung anhand von publizierten Artikeln quantitativ zu evaluieren. Einige populäre (und umstrittene) Anwendungen sind davon abgeleitete Bewertungen für Fachzeitschriften, Forscher oder Forschungseinrichtungen.

Bei dieser eher sammlungsbasierten Zitationsanalyse werden weniger einzelne Publikationen betrachtet, sondern vielmehr Mengen von Dokumenten mit bestimmten Eigenschaften. Beispiele solcher Sammlungen sind alle Dokumente eines Autors oder alle innerhalb eines bestimmten Zeitraums veröffentlichten Artikel eines wissenschaftlichen Magazins. Ein prominentes Beispiel dieser Analyseart ist der sogenannte *Impact Factor*, welcher jährlich auf Basis des *Science Citation Index* berechnet und im *Journal Citation Reports* veröffentlicht wird. Dieser Indikator gibt die durchschnittliche Anzahl von Zitierungen eines bestimmten Jahres auf Artikel der vorherigen beiden Jahre eines Journals an und wird häufig zur Bewertung der Bedeutung von Fachzeitschrift in ihrem Themenfeld herangezogen (vgl. [DLM05]). Eine kleine Auswahl solcher bibliometrischen Indikatoren werden in Abschnitt 3.4 vorgestellt.

Aus dem Bereich des *Information Retrieval* kommen eher Rankingverfahren für einzelne Dokumente. Diese haben besonders durch die Entwicklung des World Wide Webs neuen Aufschwung für die Anwendung in Suchmaschinen bekommen. Da Zitationsgraphen mit Webgraphen gewisse Ähnlichkeit haben, lassen sich dafür entwickelte Verfahren in der Regel ohne große Änderungen auch in der Zitationsanalyse benutzen. In vielen Fällen basieren die im WWW genutzten Bewertungsverfahren auf älteren Ideen aus der Bibliometrie. Generell zu beachten ist aber, dass Zitationsgraphen dennoch eine etwas andere Struktur als Webgraphen besitzen. In Abschnitt 3.5 werden einige dieser graphenbasierten Rankingverfahren aus dem Web-Mining-Bereich und ihre Benutzung auf Zitationsgraphen vorgestellt.

Die Zitationsanalyse hat noch eine Vielzahl an weiteren Anwendungsbereiche. Dazu

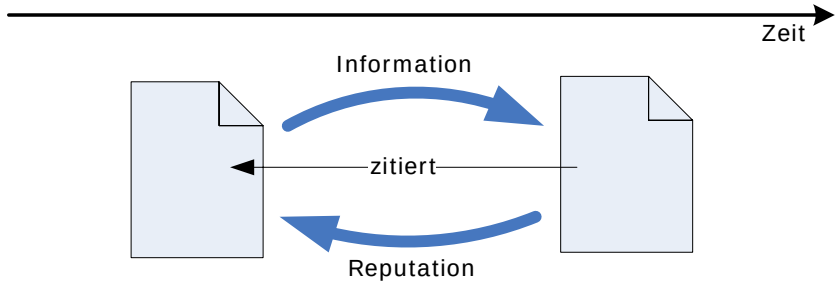


Abbildung 2.2.: Austausch von Information und Reputation zwischen zitierendem und zitiertem Dokument

gehört die Verfolgung der Entwicklung von Forschungsthemen und ihren Beziehungen untereinander. Genauso können auch die Forscher und die Beziehungen zwischen ihnen und den Themengebieten als auch untereinander untersucht werden. Hierzu stehen eine Vielzahl von unterschiedlichen Methoden aus dem Data-Mining zur Verfügung.

2.3. Die Bibliographie „Machine Learning in Strategic Game Playing“

Die zu analysierende Bibliographie „Machine Learning in Strategic Game Playing“, im Folgenden nur noch LIG genannt, ist eine von Prof. J. Fürnkranz gepflegte Sammlung von Literaturangaben zu dem Themengebiet Maschinelles Lernen in Spielen [14]. Die Bibliographie ist unter anderem im gängigen BibTeX-Format (siehe [13]) verfügbar. Ein typischer BibTeX-Eintrag aus LIG sieht beispielweise wie folgt aus:

```
@TechReport{lig*Moriarty93,
 author =       "David Moriarty and Risto Miikkulainen",
 title =        "Evolving Complex Othello Strategies
                 Using Marker-Based Genetic Encoding
                 of Neural Networks",
 institution = "Department of Computer Sciences,
                 University of Texas",
 address =      "Austin, TX",
 year =         "1993",
 number =       "AI93-206",
 LEARNING =     "Evolutionary, Neural Network",
 GAME =         "Othello",
 AVAILABLE =    "y",
 URL =          "ftp://ftp.cs.utexas.edu/pub/AI-Lab/tech-reports
    /UT-AI-TR-93-206.ps.Z"
}
```

Dieser Datensatz beschreibt einen technischen Bericht mit den üblichen bibliographischen Angaben, wie Titel, Autoren und Veröffentlichungsjahr. Neben diesen Daten gibt es in LIG zusätzliche Angaben zu den in der Publikation behandelten Spielen sowie den benutzten Lernverfahren. Etwaige Referenz- oder Zitationsinformationen der Dokumente sind dagegen nicht verzeichnet.

2.4. Zitationsdatenbanken

Die Ausgangsbibliographie LIG beinhaltet nur eingeschränkt Informationen über die enthaltenen Publikationen. Besonders fehlen Informationen über zitierte und zitierende Literatur. Selbst wenn diese Referenzinformationen vorhanden wären, würden sich diese nicht nur auf Dokumente innerhalb der Bibliographie beziehen, sondern auch auf viele weitere unbekannte Schriftstücke. Es existieren eine Reihe von Bibliographien, welche es sich zur Aufgabe gemacht haben auch die Zitationen von Publikationen zu bestimmen und als Bewertungskriterium zu nutzen. Solche Bibliographien werden dann auch Zitationsdatenbanken oder Citation Index genannt.

2.4.1. Autonomous Citation Indexing

Die Zitationsdatenbanke, welche ihren Datenbestand durch automatisches Indizieren von Dokumenten erstellen, müssen anhand der Referenzangaben im Text die passenden Einträge in der Datenbank finden. Dieser *Citation Matching* genannte Vorgang ist die Hauptaufgabe dieser Datenbanken. Die Literaturangaben müssen in den Dokumenten gefunden und extrahiert und anschließend mit anderen Zitationen verglichen werden um Übereinstimmungen zu finden. Dabei liegt das größte Problem darin, dass sehr viele verschiedene Möglichkeiten existieren um ein Dokument zu referenzieren. Die Literaturangabe kann in einer Fußnote stehen oder gesammelt am Ende der Publikation. Zusätzlich gibt es kein einheitliches Format um eine Publikation zu zitieren. Eine kleine Auswahl an real existierenden Zitationen eines Dokuments: [1]

- Moriarty, D., Miikkulainen, R. (1993). **Evolving Complex Othello Strategies Using Marker–Based Genetic Encoding of Neural Networks**. Tech. rep. AI93–206, Department of Computer Sciences, University of Texas at Austin.
- D. Moriarty, R. Miikkulainen, "**Evolving Complex Othello Strategies Using MarkerBased Genetic Encoding of Neural Networks**", Technical Report AI93-206, Department of Computer Science, University of Texas at Austin (1993).
- D. Moriarty and R. Miikkulainen, "**Evolving complex Othello strategies using marker-based genetic encoding of neural networks,**" Tech. Rep. AI93-206, The University of Texas at Austin, Sep. 1993.

[1] Entnommen aus http://citeseer.ist.psu.edu/check/74730

An diesen Beispielen sieht man, dass die Zitationen für ein Dokument sehr unterschiedlich aussehen können und keinem allgemeinen Standard folgen. Hier unterscheiden sich z. B. die Position des Publikationsjahres. Die Autorenangaben sind ebenfalls nicht identisch: Mal wird der Nachname zuerst verwendet und in anderen Fällen der Vorname (oder dessen Initialen). Im Weiteren kommen auch vereinzelt Schreibfehler oder andere Schreibweisen vor. Hier unterscheiden sich bspw. das Wort „marker-based“ in den angegebenen Titeln. In [GBL98, LBG99] werden verschiedene Lösungsansätze zu diesen Problemen beschrieben.

2.4.2. Auswahl frei verfügbarer Zitationsdatenbanken

Es gibt eine größere Anzahl an Zitationsdatenbanken, welche sich zumeist relativ speziell auf ein Themenbereich konzentrieren und Bestandteil von bibliographischen Datenbanken sind. In dieser Arbeit werden nur frei zugängliche Zitationsdatenbanken berücksichtigt, weswegen in der folgendenen Auswahl u. a. der ansonsten häufig genannte *Science Citation Index* nicht berücksichtigt wird.

CiteSeer (Scientific Literature Digital Library) [1] ist eine bibliographische Datenbank, welche sich auf Publikationen aus dem Bereich der Informatik konzentriert. Bei der Arbeit mit CiteSeer muss man genauer zwischen Dokumenten und Zitationen unterscheiden. Dokumente sind im Volltext bekannt, während eine Zitation auch ein Dokument bezeichnen kann, dessen Text selbst nicht verfügbar, aber in der Referenzangabe eines verfügbaren Dokuments gelistet ist. Ein Großteil der Meta-Daten zu den verfügbaren Dokumenten sind als gepacktes Archiv oder über das standardisierte OAI-PMH-Protokoll verfügbar. Diese Daten beinhalten auch Referenzen zu zitierten Dokumenten, sofern diese ebenfalls als Volltexte vorhanden sind. Nicht als Dokumente vorliegende Einträge lassen sich zur Zeit nur über das Web-Interface abfragen.[2] Eine genaue Beschreibung der verfügbaren Daten findet sich in Anhang C.2.1 und in Anhang D.

Cora ist eine automatisch erstellte Zitationsdatenbank, welche neben vollständigen Referenzdaten auch eine umfangreiche Kategorisierung der verzeichneten Publikationen auf Basis der Volltexte enthält [MNRS00]. Die vollständigen Datensätze sind als Archive verfügbar [16].

Digital Bibliography & Library Projekt ist eine Datenbank der Universität Trier, welche bibliografische Informationen zu den wichtigsten Journalen und Berichten aus den Bereichen der Informatik bietet (siehe [15]). Im Gegensatz zu CiteSeer und Cora enthält die Datenbank ausschließlich bibliografische Daten und nicht die zugehörigen Dokumente. Die Daten werden auch nicht automatisch extrahiert, so dass nicht mit vielen Fehlern zu rechnen ist und es in der Regel auch keine Mehrfacheinträge gibt. Die Daten sind tagesaktuell als XML-Datei verfügbar und zu einigen (wenigen) Publikationen sind auch die Zitationen verfügbar.

[2] Es existiert noch ein CiteSeer-API-Projekt unter http://labseer.ist.psu.edu/api/, welches aber scheinbar nicht mehr betrieben wird.

Google Scholar [3] beschränkt sich nicht auf bestimmte Fachgebiete und erhält seine Daten aus den unterschiedlichsten Quellen, wie zum Beispiel auch aus nicht-öffentlichen Bibliothekskatalogen. Genaue Zahlen über die Datenbankgröße werden von Google nicht genannt und die enthaltenen Einträge sind derzeit nur über das Web-Interface verfügbar.

2.5. Kritiken und Einschränkungen

Es gibt wohl keine bibliometrischen Bewertungen, welche nicht kritisiert werden. Dabei wird regelmäßig davor gewarnt, Bewertungen isoliert zu betrachten und im Besonderen auch die möglichen Fehlbeurteilungen zu bedenken.

Es gibt einige Hauptfehlerquellen, die für die Zitationsanalyse immer zu beachten sind:

- Die Grundlage der erfassten Daten. Viele Zitationsdatenbanken sind bspw. auf im WWW frei verfügbaren Publikationen aufgebaut, was zur Folge hat, dass auch nur diese einen Einfluß auf die Bewertung nehmen. Generell sollte die Fokussierung der Daten beachtet werden.
- Fehler in den Daten. Dies können zum einen unterschiedliche oder falsche Schreibweisen von Wörtern und Namen sein oder auch fehlerhafte Beziehungen zwischen den Publikationen. Ein häufiger Fehler bei ACI entsteht z. B. dadurch, dass oft nicht zwischen einem Bibliographieteil und den Referenzangaben unterschieden wird. Die Angaben in einer Bibliographie stellen gerade keine Referenzen dar, sondern nur mögliche Referenzen.
- Die Motivation der Zitierung. Diese muß nicht immer nur thematisch veranlaßt sein, sondern kann bspw. auch strategisch erfolgen um gezielt die Bewertungen anderer Publikationen zu erhöhen. Anderseits werden eher populäre Arbeiten zitiert und nicht unbedingt solche mit der größten wissenschaftlichen Relevanz, was eher auf eine Nachlässigkeit zurückzuführen ist.

3. Theorie

3.1. Duplikaterkennung

In dieser Arbeit wird davon ausgegangen, dass die Metadaten der einzelnen Dokumente sowie die Zitationsbeziehungen bereits komplett von der Zitationsdatenbank extrahiert wurden. Bei der Arbeit mit unterschiedlichen Datenbanken kann es dann nötig sein zueinander passende Dokumente aus zwei verschiedenen Sammlungen zu finden. Für ein Dokument handelt es sich also um eine Suchanfrage mit allen Metadaten eines Dokuments als Suchparameter. Theoretisch können dabei die gleichen Verfahren wie beim Citation Matching aus Abschnitt 2.4.1 benutzt werden. Im Gegensatz dazu sind die Angaben aber strukturiert, wodurch sich das Problem vereinfacht.

Der zum BibTeX-Eintrag von Seite 7 entsprechende Eintrag aus CiteSeer lautet beispielsweise:

```
@techreport{moriarty93evolving,
 author = "David Moriarty and Risto Miikkulainen",
 title =  "Evolving Complex Othello Strategies Using
           Marker-Based Genetic Encoding {ofNeural}
           Networks",
 number = "AI93-206",
 month =  "1,",
 year =   "1993",
 url =    "citeseer.ist.psu.edu/moriarty93evolving.html"
}
```

Ein Vergleich dieses Eintrags mit dem entsprechenden Eintrag aus LIG zeigt, dass sich die genannten Datenfelder unterscheiden (*institution* und *address* fehlen in CiteSeer) und zusätzlich passen auch die Werte nicht vollständig zusammen (Werte von *title* und *url* stimmen nicht überein). Um automatisch Einträge anhand der Metadaten abzugleichen, reicht daher kein direkter 1:1-Vergleich aller vorhandenen Datenfelder aus.

3.1.1. Auswahl der zu vergleichenden Datenfelder

Einige Datenfelder sind für einen Vergleich wichtiger als andere. Der Titel ist sicherlich wertvoller als ein Kommentar. Es spielt auch eine Rolle auf welcher Ebene „Gleichheit“ erzielt werden soll. Sind zwei verschiedene Auflagen eines Buches gleich?

Die Ebene der Gleichheit kann man für das in dieser Arbeit gegebene Problemgebiet recht einfach definieren: Zwei Dokumente sind gleich, wenn beide Volltexte annähernd gleich sind. Es soll also keine Rolle spielen in welcher Form etwas veröffentlicht wurde, sondern nur der Inhalt soll maßgebend sein. Da die Volltexte und auch die üblicherweise angegebenen Kurzzusammenfassungen (Abstracts) im Allgemeinen nicht verfügbar sind, scheidet der Vergleich darüber aus. Als eine quasi Ultrakurzzusammenfassung verbleiben schließlich nur noch die Dokumententitel zum Vergleich.

Die wichtigste Angabe zu einem Dokument ist damit der Titel. Aufgrund der relativen Kürze des Titels, kann es aber leicht zu Verwechslungen kommen. Hauptsächlich in Fällen mit einfachen Titeln, wie z. B. „Machine Learning“, reicht der Titel nicht aus um das Dokument eindeutig zu beschreiben. Als weitere bibliographische Angabe können dann zusätzlich noch die Autoren sowie das Publikationsjahr genutzt werden. Zumindest der Titel und die Autoren stellen die kleinste mögliche Menge an Angaben dar, um eine Publikation einigermaßen sicher zu beschreiben.

Da die Metadaten in vielen Datenbanken zum überwiegenden Teil automatisiert ermittelt wurden, sind fehlende oder fehlerhafte Daten relativ häufig und erlauben teilweise keine genaue Identifizierung der Publikation, weil beispielweise der ganze Titel fehlt. Darüber hinaus gibt es auch öfters Mehrfacheinträge (Dubletten) einer Publikation mit teils unterschiedlichen Daten.

3.1.2. Online-Suche vs. Offline-Suche

Für die verschiedenen Suchverfahren ist es sinnvoll zwischen einer Online- und einer Offline-Suche zu unterscheiden.

Eine Online-Suche erlaubt keine Vorverarbeitung der zu durchsuchenden Datenbasis resp. des Suchraums, weil dazu entweder nicht genügend Zeit oder Speicherplatz vorhanden ist. Für die Duplikatsuche bedeutet eine Online-Suche, dass es notwendig ist jeden Eintrag aus der Datenbank mit der Anfrage zu vergleichen um alle passenden Einträge zu finden.

In einer Offline-Suche wird hingegen ein dauerhafter Index angelegt, welcher über bestimmte Eigenschaften einen direkten Zugriff auf einzelne Daten erlaubt. In der Duplikatsuche könnten damit beispielsweise direkt alle Einträge mit gleichem (oder unbekanntem) Publikationsjahr ermittelt werden, welche anschließend noch genauer untersucht werden können. Weitere indizierbare Eigenschaften sind beispielsweise Autorennamen, Titelwörter oder im Titel enthaltene n-Gramme mit welchen sich die Treffermenge weiter eingrenzen läßt.

3.1.3. Boolesche Suche

Die boolesche Suche ist eine scharfe Suche. Es werden genaue Vorgaben gemacht, welche Eigenschaften die gesuchten Einträge besitzen sollen und dementsprechend auch nur

solche gefunden. Um entsprechende Einträge zu einem Dokument in einer Datenbank zu finden, könnte die Suchanfrage aus dem Titel des Dokuments mit der Vorgabe, dass alle Wörter aus der Anfrage in der korrekten Reihenfolge im Titelfeld enthalten sein müssen, bestehen.

Der Vorteil dieser Suche ist, dass sie mit einem ensprechenden Index sehr schnell und effizient als Offline-Suche implementiert werden kann. Der Nachteil der scharfen Suche ist, dass die Filterung in einigen Fällen zu restriktiv ist. Kleine Abweichungen zwischen den Wörtern führen dazu, dass die Dokumente nicht als Duplikat erkannt werden, was man beim direkten Vergleich der Titel in den beiden BibTeX-Beispiele auf den Seiten 7 und 11 erkennt: Beim ersten Eintrag heißt es „of neural“ und im zweiten „ofNeural“.

3.1.4. Matchen mit Abstandsfunktionen

Als Alternative zur Booleschen Suche kann eine Distanz zwischen zwei Dokumenten definiert werden, welche die Ähnlichkeit von zwei Dokumenteneinträgen angibt. Eine einfache Suche besteht dann aus einem Vergleich des Anfragedokuments mit jedem einzelnen Dokument in der Datenbank. Als Suchergebnis können dann die am besten passenden Dokumente, durch ihre Distanz zum Anfragedokument sortiert, angegeben werden.

Eine einfache Dokumentendistanz beschränkt sich auf eine String-Distanz der Dokumententitel. Eine String-Distanz gibt für zwei String eine Bewertung der Ähnlichkeit an. Eine Übersicht über verschiedene solcher Distanzmaße findet sich in [Nav01]. Wir betrachten hier die Levenshtein-Distanz, welche aufgrund der noch folgenden Optimierung genauer betrachtet wird.

Levenshtein-Distanz

Für einen reinen Titelvergleich bieten sich eine Reihe von Stringdistanzen an. Ein Vetreter ist die Levenshtein-Distanz, welche auch als Edit-Distanz bekannt ist. Sie mißt die minimale Anzahl der Operationen Einfügen, Löschen und Ersetzen, welche nötig sind um eine Zeichenkette in eine andere zu überführen.

Für zwei Zeichenketten $s_{1\ldots n}$ und $t_{1\ldots m}$ kann die Edit-Distanz $\mathrm{lev}(s,t)$ mit Hilfe einer Matrix $D_{0\ldots n,0\ldots m}$ berechnet werden, welche durch die Gleichungen

$$D_{i,0} = i \tag{3.1}$$

$$D_{0,j} = j \tag{3.2}$$

$$D_{i,j} = \min \begin{cases} D_{i-1,j-1} + \begin{cases} 0 & \text{, falls } s_i = t_j \\ 1 & \text{, sonst} \end{cases} \\ D_{i-1,j} + 1 \\ D_{i,j-1} + 1 \end{cases} \tag{3.3}$$

rekursiv beschrieben wird. Jeder Eintrag $D_{i,j}$ repräsentiert die Edit-Distanz zwischen den Teilstrings $s_{1...i}$ und $t_{1...j}$. Die Distanz zwischen den vollständigen Zeichenketten s und t ist demzufolge $\text{lev}(s,t) = D_{n,m}$.

Eine einfache, aber wichtige Eigenschaft der Levenshtein-Distanz ist, dass sie mindestens so groß ist, wie der Längenunterschied der beiden Zeichenketten:

$$\text{lev}(s_{1...n}, t_{1...m}) = D_{n,m} \geq |n - m| \tag{3.4}$$

Die Berechnung der Levenshtein-Distanz kann mit linearem Speicherplatzbedarf realisiert werden, da in (3.3) lediglich die vorhergehende und aktuelle Zeile benötigt wird (bei zeilenweisem Matrixaufbau). Einen möglichen Algorithmus zeigt Listing 3.1, welcher in dieser Form auch in der Klasse `net.stegi.citana.matcher.StringDistance` implementiert ist.

```
Eingabe: String s mit Länge n, String t mit Länge m
Ausgabe: Edit-Distanz

// Erste Zeile initialisieren
fuer j ← 0 bis m
    a_j ← j

fuer i ← 1 bis n {
    b_0 ← i
    fuer j ← 1 bis m {
        falls s_i = t_j
            dann   ersetzung ← 0
            sonst  ersetzung ← 1
        b_j ← min(a_j + 1, b_{j-1} + 1, a_{j-1} + ersetzung)
    }
    a ↔ b // Zeileninhalte tauschen
}
return a[m]
```

Listing 3.1: Levenshtein-Distanz

3.1.5. Einfacher unscharfer Suchalgorithmus

Es gibt durchaus verschiedene Indizierungsmethoden für *Approximate String Matching* (siehe [NBYST01]), sodass eine Offline-Suche effizient implementiert werden kann. Wir beschränken uns aber aufgrund des relativ großen Aufwands für einen solchen Index auf die Implementierung einer Online-Suche. Da die Berechnung der Levenshtein-Distanz jedoch kostenintensiv ist, wird hier versucht zumindest diese möglichst ganz zu vermeiden oder soweit wie möglich abzukürzen.

Vermeiden kann man einen Stringvergleich durch einen einfacheren booleschen Vergleich, welcher direkt in die Distanzfunktion eingebettet wird. Bei der Distanzberechnung könnte beispielsweise zuerst die Jahresangabe verglichen und bei Ungleichheit direkt eine Distanz von Unendlich zurückgeben werden. Dies funktioniert in allen Fällen, in denen für beide Einträge ein Publikationsjahr angegeben ist.

Neben der Vermeidung des Stringvergleichs kann die Berechnung der Levenshtein-Distanz abgebrochen werden, sobald die Distanz eine gesetzte Grenze nicht mehr unterschreiten kann. Zu diesem Zweck wird die Ergebnismenge in der Größe sowie in der Güte wie folgt eingeschränkt: Die Ergebnismenge soll nur Dokumente mit einer Distanz zum Anfragedokument von maximal d_{max} enthalten und möglichst maximal n_{max} Einträge enthalten. Das Ergebnis soll zusätzlich unabhängig von der Reihenfolge der Dokumente in der zu durchsuchenden Bibliographie sein. Dadurch kann der Fall eintreten, dass auch mehr als d_{max} beste Dokumente (solche mit gleicher Distanz zum Anfragedokument) gefunden werden.

Ein möglicher Alorithmus ist in Listing 3.2 gezeigt, welcher in ähnlicher Form in der Java-Klasse `net.stegi.citana.matcher.BatchMatcher` implementiert wurde. Es werden die Distanzen zwischen Anfragedokument q und jedem Dokument in $d \in B$ in $\mathrm{dist}(q, d, limit)$ berechnet, wobei der zusätzliche Parameter *limit* eine obere Grenze angibt, für die ein exaktes Ergebnis gefordert wird. Sobald bei der Distanzberechnung absehbar ist, dass die Distanz größer als *limit* ist, kann die Berechnung abgebrochen und ein beliebiger Wert größer als *limit* zurückgegeben werden. Die Schranke *limit* wird bei Bedarf nach unten angepaßt, sobald die Ergebnismenge genügend Einträge enthält. Die Distanzen $\mathrm{dist}(q, r \in R)$ brauchen natürlich nicht jedesmal neu berechnet zu werden, sondern können in der Ergebnismenge mit den zugehörigen Dokumenten gespeichert werden.

Zwei Abschätzungen bieten sich für einen vorzeitigen Abbruch an. Beide basieren darauf, dass bei der Distanzberechnung jede Matrixzeile eine untere Abschätzung der Gesamtdistanz enthält. Die erste untere Grenze ist

$$\forall i \in [0, n]. \quad \mathrm{lev}(s, t) = D_{n,m} \geq \min_{j \in [0,m]} (D_{i,j})\,. \tag{3.5}$$

Beweis. Für ein festes, aber beliebiges $i \in [0, n]$ gilt:

$$\begin{aligned} \mathrm{lev}(s, t) &= \min_{j \in [1,m]} (\mathrm{lev}(s_{1\ldots i}, t_{1\ldots j}) + \mathrm{lev}(s_{i+1\ldots n}, t_{j+1\ldots m})) \\ &\geq \min_{j \in [1,m]} (\mathrm{lev}(s_{1\ldots i}, t_{1\ldots j}) \end{aligned}$$

Etwas informaler kann der Beweis auch einfach aus Listing 3.1 abgeleitet werden. Die Fälle $i = 0$ und $i = n$ sind trivial. Für jede berechnete neue Zeile $0 < i < n$ wird im besten Fall ein Wert aus der vorherigen Zeile $i - 1$ direkt oder im schlechteren Fall mit Eins addiert übernommen. Alternativ kann noch der Wert in der vorherigen Spalte plus Eins übernommen werden, welcher aber selbst spätestens in der nullten Spalte wieder um Eins höher ist als der entsprechende Wert in der vorherigen Zeile. Jede neue Zeile

```
Eingabe: Anfragedokument q, Bibliographie B, n_max, d_max
Ausgabe: Ergebnismenge R ⊂ B

// Initialisierung
R ← ∅
limit ← d_max

fuer jedes d ∈ B {
    falls dist(q, d, limit) < limit tue {
        R ← R ∪ {d}
        falls |R| > n_max und min{dist(q, r ∈ R)} ≠ max{dist(q, r ∈ R)}{
            R ← R\{arg max_{r∈R} dist(q, r)}
            limit ← max{dist(q, r ∈ R)})
        }
    }
}
```

Listing 3.2: Dokumenten-Matching

kann die minimale Distanz der vorherigen also nur erhöhen. Damit gilt insbesondere auch (3.5). □

Die zweite Abschätzung ist eine Verschärfung von (3.5):

$$\forall i \in [0, n]. \quad \mathrm{lev}(s, t) = D_{n,m} \geq \min_{j \in [0,m]} \left(D_{i,j} + |n - m - i + j| \right). \tag{3.6}$$

Beweis. Die Verschärfung ergibt sich aus (3.4), als der Eigenschaft, dass die Edit-Distanz zwischen zwei String mindestens so groß sein muss, wie der Längenunterschied zwischen diesen. Damit beträgt die Distanz der Teilstrings $s_{i+1\ldots n}$ und $t_{j+1\ldots m}$ mindestens $|(n - i + 1) - (m - j + 1)| = |(n - i) - (m - j)|$. Diese Restdistanz kann direkt in die Abschätzung (3.5) eingesetzt werden und ergibt damit die untere Abschätzung (3.6).

$$\begin{aligned} \mathrm{lev}(s, t) &= \min_{j \in [1,m]} (\mathrm{lev}(s_{1\ldots i}, t_{1\ldots j}) + \mathrm{lev}(s_{i+1\ldots n}, t_{j+1\ldots m})) \\ &\geq \min_{j \in [1,m]} (\mathrm{lev}(s_{1\ldots i}, t_{1\ldots j}) + |(n - i) - (m - j)|) \end{aligned}$$

□

Durch die Berechnung und Überprüfung der Abbruchbedingung wird die Distanzberechnung selbst komplexer. Der Aufwand lohnt sich nur, falls *limit* anfangs bereits entsprechend klein ist oder im Laufe des Algorithmus sehr schnell klein wird, sodass in der Mehrzahl der Fälle früh genug abgebrochen werden kann. Verschiedene implementierte Dokumentendistanzen sowie deren Laufzeitverhalten werden in Anhang B untersucht.

3.2. Zitationsgraphen

Für eine Menge von Dokumenten und deren Zitationsbeziehungen lässt sich ein Graph $G = (V, E)$ erstellen, in welchem die Dokumente die Knoten V bilden und die Zitierungen durch gerichtete Kanten $E \subset V \times V$ repräsentiert werden. Ein Beispiel eines solchen Graphen ist in Abbildung 3.1 dargestellt.

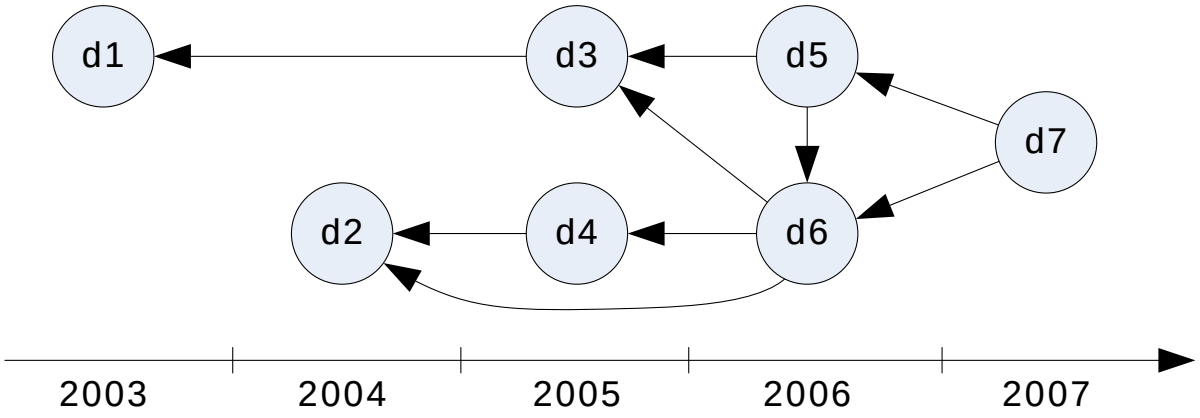


Abbildung 3.1.: Ein einfacher Zitationsgraph

Eine Besonderheit bei Zitationsgraphen ist, dass die Kanten in der Zeit gerichtet sind, was direkt aus der zeitlichen Richtung der Referenzen (vgl. Abschnitt 2.1) folgt. Aus diesem Grund sind Zitationsgraphen in der Regel azyklisch. Dies unterscheidet Zitationsgraphen auch hauptsächlich von Webgraphen, deren Dokumente sich auch nach der Veröffentlichung noch ändern können und dadurch Zyklen entstehen.

3.2.1. Skalenfreie Netzwerke

Nicht alle Knoten in einem Graph müssen die gleiche Anzahl von Kanten (Kantengrad) besitzen. Die Bandbreite der Kantengrade in einem Graphen ist durch die Verteilungsfunktion $P(k)$ charakterisiert. Diese Funktion gibt die Wahrscheinlichkeit an, dass ein zufällig ausgewählter Knoten genau k Kanten besitzt. Normale Zufallsgraphen werden erzeugt indem zufällige Kanten hinzugefügt werden. In diesem Fall werden die meisten Knoten in etwa die gleiche Anzahl von Kanten besitzen, welche in etwa die durchschnittliche Anzahl von Kanten pro Knoten sein wird. Genauer sind die Kantengrade in einem Zufallsgraph eine Poisson-Verteilung, welche ihr Maximum an der Stelle $\langle k \rangle$ einnimmt (vgl. [AB02]).

In skalenfreien oder skaleninvarianten Graphen ist diese Eigenschaft nicht mehr vorhanden: Es gibt gerade keine typische Anzahl von Kanten. Die Verteilung der Kantengrade folgt hier einem Potenzgesetz $P(k) \sim k^{-\gamma}$. Der Wert für γ variiert je nach Graph und liegt in den meisten Fällen zwischen 1 und 3 (vgl. [AB02]).

Diese Invarianz bezüglich der Kantengrade kann bei einer Vielzahl von realen Netzwerken beobachtet werden und in der Regel ebenso in Zitationsgraphen. Da hier gerichtete Kanten vorkommen, kann man auch den Kantengrad eines Knoten nach eingehenden und ausgehenden Kanten unterscheiden. In Abschnitt 5.1.1 werden diese Kantengradverteilungen für verschiedene Zitationsdatenbanken genauer untersucht. Vorausblickend kann gesagt werden, dass Zitationsgraphen bezüglich der Zitationen, also der eingehenden Kantenanzahl, ein skalenfreies Netzwerk bildet. Konkret heisst dies, dass es einige sehr häufig zitierte Dokumente gibt, aber die überwiegende Mehrheit der Dokumente nur relativ wenig zitiert werden.

3.2.2. Gewichtete Zitationsgraphen

Traditionell sind die Kanten von Zitationsgraphen ungewichtete Einfachkanten (vgl. [ER90]). Es spricht aber nichts dagegen, auch Zitationsgraphen mit gewichteten Kanten zu konstruieren und auszuwerten. Die später vorgestellten Algorithmen HITS, PageRank und PHITS lassen sich ohne großen Aufwand für solche Graphen anpassen oder erweitern. Allgemein soll gelten, dass das Gewicht $w(e) \in \mathbb{R}^+$ einer Kante $e \in E$ die Beziehungsstärke repräsentiert, also ein hohes Gewicht „besser“ ist.

Als einfache Gewichtung könnte, falls bekannt, die Häufigkeit der Zitierung zwischen zwei Dokumenten benutzt werden.[1] Solche Informationen sind z. B. im Cora-Datensatz enthalten (vgl. Abschnitt 5.2.1).

Die Kanten können auch aufgrund der Referenzenanzahl des zitierenden Dokuments gewichtet werden. Eine solche Gewichtung wird beispielsweise implizit im noch folgendem PageRank-Verfahren umgesetzt.

3.2.3. Adjazenzmatrix

Eine mathematisch kompakte Darstellung eines Graphen liefert die Adjazenzmatrix. Ein Graph mit n Knoten wird durch eine $n \times n$-Matrix dargestellt, indem die Spalten- und Reihenindizes der Matrix die Knoten repräsentieren und die Werte an den entsprechenden Positionen die Beziehung zwischen den Knoten, also die Kanten. Für einen gewöhnlichen Zitationsgraphen ist die Zitationsmatrix $\mathbf{M} = (m_{i,j})_{n \times n}$ also wie folgt definiert:

$$m_{i,j} = \begin{cases} 1, & \text{falls } i\text{-tes Dokument das } j\text{-te Dokument zitiert,} \\ 0, & \text{sonst.} \end{cases} \tag{3.7}$$

Für gewichtete Graphen gilt allgemein, dass die Einträge in $\mathbf{M}$ dem Kantengewicht $w_{i,j}$ entsprechen. Nicht vorhandene Kanten haben per Definition ein Gewicht von 0. Als

[1] Aus graphentheoretischer Sicht handelt es sich hierbei eigentlich um Mehrfachkanten. Der Einfachheit halber werden wir Mehrfachkanten ebenfalls als Gewichte bezeichnen und als solche behandeln.

Beispiel wird der Graph aus Abbildung 3.1 durch folgende Matrix beschrieben:

$$\mathbf{M} = \begin{array}{c} \\ d1 \\ d2 \\ d3 \\ d4 \\ d5 \\ d6 \\ d7 \end{array} \begin{array}{c} \begin{array}{ccccccc} d1 & d2 & d3 & d4 & d5 & d6 & d7 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 \end{pmatrix} \end{array}$$

An diesem Beispiel sieht man sofort das Hauptproblem dieser Darstellung: Der überwiegende Anteil der Matrix wird nicht benutzt. Allein aufgrund der zeitlich gerichteten Kanten muss selbst im Extremfall mehr als die Hälfte der Einträge einer Zitationsmatrix Nullen enthalten. In der Praxis ist der wirklich genutzte Anteil noch weitaus geringer. Neben dem großen Platzbedarf müssen zusätzlich auch die mit dieser Zitationsmatrix arbeitenden Algorithmen unnötigerweise jeden einzelnen Null-Wert in die Berechnungen einbeziehen. Daher werden bei der Speicherung der Graphen und bei der Implementierung der Algorithmen nur vorhandene Kanten, also Kanten mit einem Gewicht größer als null, berücksichtigt.

3.2.4. Abgeleitete Graphenstrukturen

Inverser Zitationsgraph

Neben dem normalen Zitationsgraphen gibt es weitere Graphen, welche man auf Basis von Zitationen erstellen kann. Die einfachste Variante ist der inverse Zitationsgraph G^{-1}, in welcher die „zitiert"-Beziehung durch die „wird-zitiert-von"-Beziehung getauscht wird. Dies entspricht als Matrixoperation der transponierten Zitationsmatrix:

$$\mathbf{M}_{G^{-1}} = \mathbf{M}_G^T \tag{3.8}$$

Ungerichteter Zitationsgraph

Aus einem gerichteten Graphen $G = (V, E)$ folgt der zugeordnete ungerichtete Graph $\underline{G} = (V, E \cup E^{-1})$, welcher für jede gerichtete Kante auch eine rückwärts gerichtete Kante enthält. Die dazu korrespondierende Matrixoperation ist:

$$\mathbf{M}_{\underline{G}} = \mathbf{M}_G + \mathbf{M}_G^T \tag{3.9}$$

Diese Matrixoperation verdoppelt bereits vorhandene symmetrische Kanten und Schleifen, sodass die Entsprechung nur für azyklisch gerichtete Graphen gilt. Wie bereits bekannt, sind diese Eigenschaften für Zitationsgraphen in der Regel gegegeben.

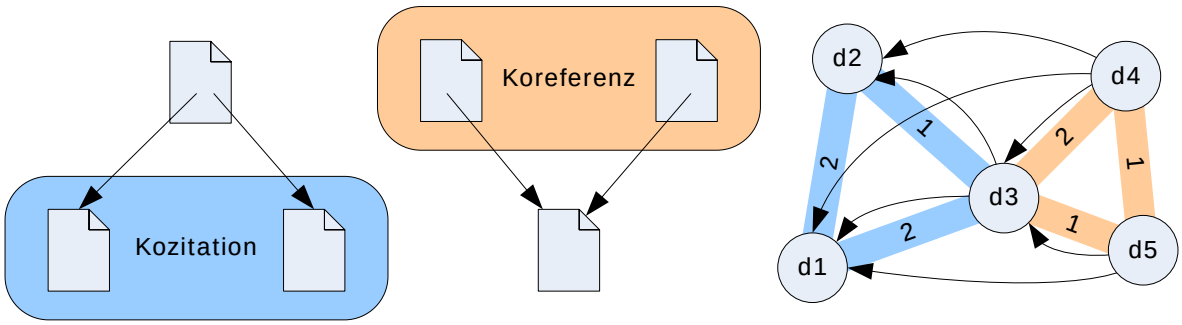


Abbildung 3.2.: Von der „zitiert“-Beziehung (Pfeile) sind abgeleitetet die Kozitation und die Koreferenz (Kasten) mit deren Beziehungsstärken.

Kozitation

Eine besondere Stellung in der Bibliometrie stellt die Kozitation dar (vgl. Abbildung 3.2). Bei der Kozitation besteht eine symmetrische Beziehungen zwischen zwei verschiedenen Dokumenten, wenn beide von einem Dokument gemeinsam zitiert werden; sie werden kozitiert. Es kann zusätzlich die Stärke dieser Beziehung angegeben werden. Die Kozitationsstärke zwischen zwei verschiedenen Dokumenten ist die Häufigkeit in der beide Dokumente gemeinsam referenziert werden. Die Reflexivität wird häufig per Definition ausgeschlossen, sodass sich ein Dokument nicht selbst kozitiert. Wir wollen hier aber erstmal auch diese einbeziehen und später bei Bedarf genauer differenzieren.

Berechnen läßt sich die Kozitationmatrix $\mathbf{C}$ als Matrixoperation für einen durch $\mathbf{M}$ beschriebenen Graph wie folgt:

$$\mathbf{C} = \mathbf{M}^T\mathbf{M}. \tag{3.10}$$

Die so gebildete Kozitationsmatrix $\mathbf{C}$ ist im weiteren symmetrisch und in der Regel nicht transitiv.

Beispiel Als Beispiel wird die Kozitationsmatrix $\mathbf{C}$ des Zitationsgraphen aus Abbildung 3.2 berechnet. Die Zitationsmatrix $\mathbf{M}$ kann angegeben werden als:

$$\mathbf{M} = \begin{array}{c} \\ d1 \\ d2 \\ d3 \\ d4 \\ d5 \end{array} \begin{array}{c} \begin{array}{ccccc} d1 & d2 & d3 & d4 & d5 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

Aus dieser berechnet sich schließlich die Zitationsmatrix

$$\mathbf{C} = \mathbf{M}^T\mathbf{M} = \begin{pmatrix} 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 3 & 2 & 2 & 0 & 0 \\ 2 & 2 & 1 & 0 & 0 \\ 2 & 1 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Die Hauptdiagonale von $\mathbf{M}^T\mathbf{M}$ enthält hier die Anzahl aller eingehenden Kanten eines Knoten. Im allgemeinen Fall enthält sie die Summen der quadrierten Eingangsgewichte. Diese Einträge stellen reflexiven Kanten (Schleifen) des Kozitationsgraphen dar, welche häufig entfernt werden und in Abbildung 3.2 nicht dargestellt sind.

Koreferenz oder Bibliographische Kopplung

Analog zur Kozitation besteht bei der Koreferenz eine symmetrische Beziehung zwischen zwei unterschiedlichen Dokumenten, welche das gleiche Dokumente zitieren (vgl. Abbildung 3.2). Die Koreferenz ist in der Zitationsanalyse besser bekannt als bibliographische Kopplung. Um die Analogie zur Kozitation zu betonen, bleiben wir hier bei der Bezeichnung Koreferenz. Die Koreferenzstärke kann über Anzahl der übereinstimmenden Referenzen bestimmt werden.

Berechnen läßt sich die Koreferenz als Matrixoperation für einen durch $\mathbf{M}$ beschriebenen Graph wie folgt:

$$\mathbf{R} = \mathbf{M}\mathbf{M}^T \tag{3.11}$$

Hier besteht die Hauptdiagonale von $\mathbf{M}\mathbf{M}^T$ aus den Gewichten aller ausgehenden Kanten eines Knoten. Die so gebildete Koreferenzmatrix $\mathbf{R}$ ist wieder symmetrisch und in der Regel nicht transitiv.

Beispiel Die Koreferenzmatrix $\mathbf{R}$ des Zitationsgraphs aus Abbildung 3.2 berechnet sich damit zu:

$$\mathbf{R} = \mathbf{M}\mathbf{M}^T = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 2 & 1 \\ 0 & 0 & 2 & 3 & 2 \\ 0 & 0 & 1 & 2 & 2 \end{pmatrix}$$

Hier bestehen die Einträge der Hauptdiagonale aus der Anzahl aller ausgehenden Kanten der Knoten. Im Allgemeinen besteht sie aus der Summe der quadrierten Ausgangsgewichte. Diese stellen reflexiven Kanten (Schleifen) des Koreferenzgraphen dar, welche ebenfalls in Abbildung 3.2 nicht dargestellt sind.

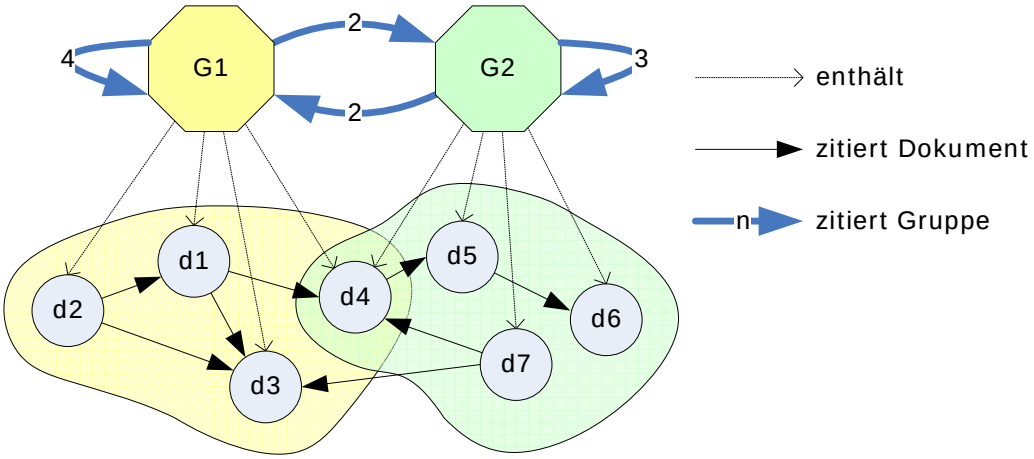


Abbildung 3.3.: Von Dokumentenbeziehungen abgeleitete Gruppenbeziehungen zwischen zwei verschiedenen, sich überlappenden Sammlungen G1 und G2.

Gruppengraphen

Als weitere Option ist es schließlich noch möglich ganze Sammlungen als einzelne Knoten zu repräsentieren. Die „zitiert“-Beziehungen zwischen diesen Gruppen ist dann die Zusammenfassung der „zitiert“-Beziehungen zwischen den enthaltenen Dokumenten. Da sich Sammlungen auch überlappen können, sofern man dies nicht per Definition ausschließt, kann es zu Verdopplung von Kanten kommen. In Abbildung 3.3 ist ein solcher Fall dargestellt. Im eigentlichen Zitationsgraph sind acht gerichtete Kanten. Die Summe der Gruppenzitierungen ist aber elf, da die drei Kanten, welche das Dokument d4 berühren, für jede Sammlungen zählen. An diesem Beispiel sieht man auch sofort, dass die sich ergebende Graphenstruktur nichts mehr mit einem normalen Zitationsgraphen gemeinsam hat. Die zeitliche Orientierung entfällt in der Regel und Schleifen, Zyklen und Mehrfachkanten sind zu erwarten. Anhand einem solchen Graph können Beziehungen zwischen Autoren, Zeitschriften, etc. untersucht werden. Auf Dokumentenebene stellt dies ein Hypergraph dar und ist damit eine Mischform von sammlungs- und graphenbasierter Ansicht.

3.3. Thematische Zitationsgraphen

Im vorherigen Abschnitt wurden Zitationsgraphen und davon abgeleitete Strukturen vorgestellt. Es wurde aber noch nicht genauer darauf eingegangen welche Dokumente überhaupt benutzt werden sollen um einen Zitationsgraphen aufzubauen und unter welchen Gesichtspunkten er schließlich analysiert werden soll. Um dies genauer betrachten zu können führen wir hier den Begriff *Thema* ein, welcher dann auch zu einer Unterscheidung der verschiedenen Analyseverfahren benutzt wird.

3.3.1. Themenbegriff

Üblicherweise wird in der Literatur der Themenbegriff für die inhaltliche Beschreibung eines Dokumentes benutzt. Es geht also um den behandelten Gegenstand oder den Grundgedanken des Textes. Eine Eigenschaft eines Dokumentes ist allgemeiner gehalten und beschreibt damit auch die äußere Form des Textes, wie z. B. die Autoren. Während eine äußere Eigenschaft zumeist recht klar beschrieben werden kann, ist dies für den Inhalt nicht so einfach. Hier behilft man sich dann zumeist mit Stichwörtern oder verschiedenen Klassifizierungen, welche wir hier aber nicht voraus setzen können.

In dieser Arbeit soll das Thema oder das Themengebiet einer Menge von Dokumenten allgemein als die Gemeinsamkeiten der enthaltenen Dokumente definiert sein. In den wenigsten Fällen wird man ein Thema direkt beschreiben können, sodass als Eratz die Dokumentenmenge selbst für das Thema stehen kann.

3.3.2. Unterteilung der Analyseverfahren

Um die Unterschiede und Zusammenhänge klarzustellen wird hier versucht die verschiedenen Verfahren in einer mathematischen Form zu beschreiben. Dabei gehen wir von einem absoluten Zitationsgraphen Z aus, welcher in der Realität aus einer durchaus begrenzten Zitationsdatenbank besteht. Genauer betrachtet wird hier ein Thema $T \subseteq Z$. Die im Folgenden benutzte Notation $r(x|y)$ soll in Anlehnung an die bedingten Wahrscheinlichkeiten ein bedingtes Ranking ausdrücken, d. h. ein Ranking von x unter dem Gesichtspunkt y. Wir unterscheiden in dieser Arbeit die folgenden drei Rankingtypen:

1. Sammlungsbasierte Zitationsanalyse stellt eine Rangordnung für verschiedenen Unterthemen oder generell für verschiedenen Dokumentensammlungen auf. Für verschiedene Sammlungen $G_i \subset T$ wird also eine Bewertungsfunktion $r(G_i|T) \in \mathbb{R}$ benutzt um damit eine Rangordnung der Sammlungen bezüglich des Themas T zu bilden. Diese Sammlungsbewertungen werden traditionell auch bibliometrische Indikatoren genannt.
2. Monothematische Zitationsanalyse stellt eine Rangordnung für verschiedene Dokumente bezüglich eines bestimmten Themengebiets oder dessen Hauptthemas auf. Es werden Bewertungen für die im Thema T enthaltenen Dokumente $d_i \in T$ unter Bezugnahme von T berechnet, also $r(d_i|T) \in \mathbb{R}$. Diese Betrachtung stellt im Grunde ein Spezialfall der Sammlungsbasierten Analyse dar in der jede Sammlung aus einem Einzeldokument besteht $G_i = \{d_i\}$.
3. Polythematische Zitationsanalyse stellt mehrere Rangordnungen für die Dokumente eines Themengebiets bezüglich einer Anzahl von Unterthemen auf, welche selbst konkret noch nicht bekannt sind. Hier geht es neben einem Ranking also auch um das Finden von Unterthemengebieten. Ausgehend von dem Themengebiet T werden Teilgebiete t_j mit ihren zugehörenden Rankings $r(d_i|t_j) \in \mathbb{R}$ für Dokumente $d_i \in T$ gesucht. Die Unterthemengebiete t_j stehen dabei für abstrakte Themen und stellen nicht unbedingt konkret einzelne Dokumentenmengen dar.

Diese Aufteilung ist nicht vollständig. Hier werden im Besonderen nur Dokumente oder Sammlungen aus T selbst betrachtet. Allgemeiner könnten aber auch beliebige Dokumente oder Sammlungen bezüglich eines Themas bewertet werden und damit speziell auch $d \in Z \backslash T$.

3.3.3. Graphen- oder Themenerweiterung

Auf Basis einer thematischen Dokumentenmenge lassen sich abhängig vom Analyseziel sowie dem Analyseverfahren auf verschiedene Arten Teilgraphen erstellen. Ausgangspunkt ist dabei der induzierte Teilgraph (Untergraph), welcher von der Dokumentenmenge gebildet wird. Ein solcher Graph besteht nur aus in der Teilmenge enthaltenen Dokumente als Knoten und allen vorhandenen „Zitiert“-Beziehungen als Kanten zwischen diesen.

Man unterstellt allgemein, dass eine Zitierung aufgrund eines thematischen Zusammenhangs gemacht wird. Damit besteht auch eine thematische Beziehung zwischen zitierenden und zitierten Dokumenten. Um Themengebiete zu finden, macht man sich zu nutze, dass thematisch ähnliche Dokumente im Zitationsgraph stärker untereinander vernetzt sind und so Cluster bilden.

Für eine Menge von Dokumenten kann man nun den induzierten Teilgraph erweitert, indem man in Beziehung stehende Dokumente hinzufügt. Solche Dokumente können direkt die Referenzen und Zitationen sein, aber auch Koreferenzen oder Kozitationen sind überlegenswerte Varianten um thematisch ähnliche Dokumente einzubeziehen. Erweitert man den Graphen jedoch mit Kozitation oder Koreferenzen, muß beachtet werden, dass die neuen Dokumente nicht zwingend mit anderen Dokumenten der Menge in einer „zitiert“-Beziehung stehen und so nicht zwangsweise ein zusammenhängender Zitationsgraph entsteht.

Für einige Verfahren können indirekt Koreferenzen und Kozitationen zwischen Knoten eine Rolle spielen. Hier sind also evtl. auch Vorgänger und Nachfolger für die Berechnung interessant, welche selbst nicht in der betrachteten Dokumentensammlung vorhanden sein sollen. Dazu bietet es sich an, solche Vorgänger- oder Nachfolgeknoten nur temporär in die Berechnung einzubeziehen und anschließend wieder zu entfernen.

Für die Graphenauswertung ist zu beachten, dass jegliche Änderung des Ursprungsgraphen durch Hinzunahme oder Wegnahme von Knoten eine Änderung der abgedeckten Themengebiete zur Folge hat und damit auch die Ergebnisse thematisch beeinflussen kann.

3.4. Bibliometrische Indikatoren

Um den „Wert“ von Dokumentensammlungen vergleichen zu können, bedarf es eines Qualitätsmaßes mit dem eine Rangliste der Sammlungen erstellt werden kann. Da der

Wert einer Sammlung im Besonderen von der Aufgabenstellung abhängt, gibt es eine Vielzahl an vorgeschlagenen und benutzten bibliometrischer Indikatoren. Diese Indikatoren messen in der Regel, wie auch die graphenbasierten Verfahren, den Zitationen einen besonderen Wert zu. Im Gegensatz zu den netzwerkbasierenden Analyseverfahren spielt hier oft weniger die konkrete Verflechtung der Gruppen untereinander eine Rolle. Wie wir am Gruppengraphen gesehen haben, sind aber diese ebenfalls durchaus als Grundlage für Indikatoren nutzbar.

Wir beschränken uns hier auf eine kleine Auswahl von einfachen bibliometrischen Indikatoren für Dokumentensammlungen $G_i \subseteq T$ aus einem Themengebiet T. Die Bedeutung der Bewertungen ist natürlich abhängig von den Kriterien aus denen die Sammlungen erstellt wurden. Anhand einer konkreten Anwendung sollen die Indikatoren vorgestellt werden: Das Thema T soll alle Dokumente aus einem bestimmten Fachgebiet und Zeitraum enthalten. Die Dokumente aus T werden schließlich nach ihren Autoren gruppiert und bilden so die (möglicherweise überlappende) Sammlungen G_i.

3.4.1. Sammlungsgrößen und Zitierraten

Der einfachste Indikator ist die **Sammlungsgröße** $n(G_i) = |G_i|$, also die absolute Anzahl der enthaltenen Dokumente. Für die Einzelautoren wäre dies eine sehr einfache Bewertung des wissenschaftlichen Outputs. Hierbei wird aber die Bedeutung und Einfluß der Dokumente nicht weiter beachtet.

Ein Indikator zur Messung des Einfluß ist die **absolute Zitierrate** aller in G_i enthaltenen Dokumente, also bspw. $c_{tot}(G_i) = \sum_{d \in G_i} c(d)$ mit $c(d)$ als Zitierrate von d. Dieser Indikator wird bei der Betrachtung von Autoren häufig eingeschränkt, indem nur Zitationen außerhalb der betrachteten Gruppe zählen. Dadurch soll eine Manipulation durch Selbstzitierungen verhindert werden.

Die absolute Zitierrate für eine Sammlung ist stark von der Größe der Sammlung abhängig. Naturgemäß werden ältere Dokumente auch mehr Zitationen haben als gerade erst erschienene, sodass hier auch ältere Dokumente bevorzugt werden. Verbessern kann man dies durch die **relative Zitierrate** $c_{rel}(G_i) = c_{tot}(G_i)/|G_i|$. Dieser Wert bestraft nun aber große Sammlungen und ist ebenfalls altersabhängig.

Als weiteren Indikator kann man die **Anzahl der signifikanten Dokumente** in G_i zählen. Ein signifikantes Dokument kann man bspw. als ein Dokument mit einer Mindestzitierrate $c(d) \geq x$ definieren. Durch die beliebige Wahl der Schranke x werden hierbei aber zufällig einzelne Sammlungen belohnt oder bestraft.

3.4.2. h-Index

In [Hir05] wird der h-Index als ein spezieller Indikator zur Beurteilung der wissenschaftlichen Leistung von Forschern auf Basis deren veröffentlichten Arbeiten vorgeschlagen. Im Besonderen sollen die Nachteile der obigen Indikatoren umgangen werden.

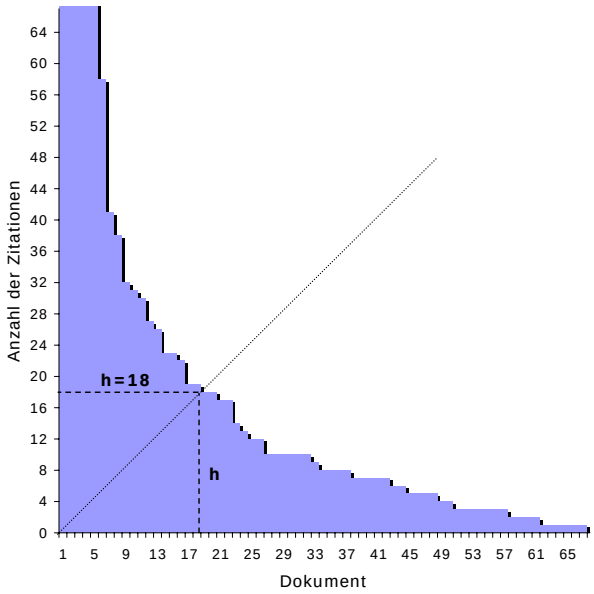


Abbildung 3.4.: Bestimmung des h-Index über die geordnete Häufigkeit der Zitationen in einer Dokumentensammlung.[2]

Ein Forscher hat demnach einen Index h, wenn h von seinen insgesamt n Veröffentlichungen mindestens jeweils h Zitationen haben und die restlichen $(n-h)$ Publikationen weniger als h Zitationen.

Der h-Index läßt sich recht einfach bestimmen, indem man die Dokumente bezüglich ihren Zitierraten absteigen sortiert. In Abbildung 3.4 ist dies für eine konkrete Sammlung gezeigt. Das größte h, für welches das h-te Dokument mehr als h oder genau h Zitationen besitzt, ist der gesuchte h-Index.

Um einen hohen h-Wert zu erhalten, muß die Sammlung also viele Dokumente enthalten und diese müssen zusätzlich auch entsprechend häufig zitiert werden. Dadurch fallen im Besonderen Einzelleistungen aus der Bewertung heraus: Ein einzelnes vielzitiertes Dokument hat keinen großen Einfluß auf die Gesamtbewertung.

Im Weiteren kann direkt aus Abbildung 3.4 entnommen werden, dass der Wert h^2 eine untere Abschätzung für die gesamte Zitationsanzahl ist. Der a-Index gibt das Verhältnis zwischen h^2 mit $h > 0$ und der Gesamtzitierrate c_{tot} durch $a = c_{tot}/h^2$ an. Wird der a-Index als eine Zweitbewertung für Sammlungen mit gleichem h-Index benutzt, entspricht dieser gerade einer Bewertung nach der Gesamtzitierrate.

[2]Publikationen von A. Gupta aus Cora-Datensatz

3.5. Monothematische graphenbasierte Bewertungen

Wir beschränken uns hier auf graphenbasierten Rankingverfahren, welche allein auf (evtl. kantengewichtete) Zitationsgraphen angewendet werden. Die hauptsächlich betrachteten Eigenschaften der Dokumente sind die Beziehungen und Beziehungsstärken zu anderen Dokumenten innerhalb des Graphen oder des so beschriebenen Themengebietes.

3.5.1. Indegree- und Outdegree-Rank

Das einfachste Ranking ist die Zählung der Zitationen oder Referenzen eines Dokumentes. Dabei kann man eine globale und eine thematische Variante unterscheiden.

Der globale Indegree besteht aus der Anzahl aller vorhandenen Zitationen einer Publikation. Diesen Wert wird man praktisch nicht ermitteln können und stattdessen die Anzahl aller vorhandenen Zitationen der Publikation in der benutzten Zitationsdatenbank benutzen. Daraus folgt aber wieder eine thematische Einschränkung, sodass man entsprechend vorsichtig mit diesen Ergebnissen sein sollte.

Der globale Outdegree besteht einfach nur aus der Anzahl aller Referenzen einer Publikation. Dieser Wert ist in den meisten Datenbanken verfügbar, aber meist nicht sehr aussagekräftig zur Beurteilung eines Dokumentes.

Der thematische Indegree zählt die Anzahl der Zitationen einer Publikation, welche in der Themenmenge enthalten sind. Dies entspricht genau der Anzahl der eingehenden Kanten. Für einen gewichteten Zitationsgraphen können wir auch entsprechend die Eingangsgewichte als die Summen aller eingehenden Kanten angeben. Als Matrixfunktion entspricht dies

$$\mathbf{w}_{in} = \mathbf{M}^T \cdot \mathbf{1}. \tag{3.12}$$

Der thematische Outdegree zählt analog zum Indegree die ausgehenden Kanten. Die Ausgangsgewichte können als die Summen der ausgehenenden Kantengewichte angegeben werden. Als Matrixfunktion folgt daraus

$$\mathbf{w}_{out} = \mathbf{M} \cdot \mathbf{1}. \tag{3.13}$$

3.5.2. PageRank

Der PageRank-Algorithmus ist ein weiteres Verfahren um eine Menge von verbundenen Dokumente allein anhand ihrer Graphenstruktur zu bewerten und wird in [PBMW98] hauptsächlich für eine Anwendung in Web-Graphen beschrieben. Das Prinzip auf dem der PageRank basiert, lautet: Der Wert eines Dokuments ist umso höher, je mehr Zitationen dazu exisitieren und je höher der Wert der zitierenden Dokumente sind.

Für einen Knoten i wird der PageRank r_i durch folgende Gleichung rekursiv beschrieben:

$$r_i = \frac{1-d}{N} + d \sum_{\forall j.(j,i)\in E} \frac{r_j}{C_j}. \tag{3.14}$$

Hierbei gibt N die Anzahl aller Knoten an, d ist ein Dämpfungsparameter und C_j ist die Anzahl der ausgehenden Links des Knoten j.

Diese Formel wird häufig mit dem Random-Surfer-Modell erläutert: Ein Web-Surfer klickt mit der Wahrscheinlichkeit von d einen beliebigen Link auf der Seite an auf welcher er sich gerade befindet und bekommt so eine neue Seite zu sehen. Mit der Wahrscheinlichkeit $1-d$ wird dem Surfer langweilig und er klickt nicht einen der gerade verfügbaren Links an, sondern wechselt auf eine komplett zufällige Seite. Der PageRank gibt dann für jede Seite die Wahrscheinlichkeit an, mit der der Random-Surfer auf diese Seite gerät.

In [PBMW98] wird ein einfacher Algorithmus (siehe Listing 3.3) angegeben um eine PageRank entsprechende Bewertung (nicht zwangsläufig eine Wahrscheinlichkeitsverteilung) für alle Knoten zu berechnen. Dabei ist die Übergangsmatrix $\mathbf{A}$ wie folgt definiert:

$$A_{i,j} = \begin{cases} 1/C_i, & \text{falls } i\text{-tes Dokument das } j\text{-te Dokument zitiert,} \\ 0, & \text{sonst.} \end{cases} \tag{3.15}$$

Wir betrachten hier Zitationsgraphen deren Referenzen in den seltensten Fälle alle im Graph selbst enthalten sind. Um den Wert einer ausgehenden Kante wirklichkeitsnäher zu gewichten, sollte man die vom Ziel unabhängige Anzahl Out_j^G der ausgehenden Links als Parameter C_i zur Linkgewichtung benutzen. Die daraus folgende Werteweitergabe ist in Abbildung 3.5 skizziert, wo von d3 an d5 nur ein Drittel des PageRanks weitergegeben wird, obwohl im Zitationsgraph d5 die einzige Referenz von d3 darstellt. Im Sinne des Random-Surfer-Modell hat diese Änderung zur Folge, dass der Surfer wieder wahllos allen Links (oder hier Referenzen) folgt, auch solche die im Zitationsgraph selbst nicht enthalten sind, weil das Zieldokument thematisch nicht passt. Sobald er bemerkt, dass er das Thema oder die betrachtete Dokumentenmenge verlassen hat, springt er wieder zufällig zu einem darin enthaltenen Dokument.

Ein weiterer Parameter des in Listing 3.3 angegebenen Algorithmus ist der Quellen-Vektor $\mathbf{s}$. Im Random-Surfer-Modell gibt dieser Parameter die Dokumente für einen zufälligen Sprung an. Für das normale PageRank-Verfahren ist $\mathbf{s} = \alpha \cdot \mathbf{1}$ einfach eine gleichmäßige Verteilung über alle Dokumente mit einem vorgegeben α (bspw. $\alpha = 0{,}15$). Diese Verteilung kann aber auch abgeändert werden indem nur bestimmte Dokumente darin ein Gewicht bekommen und somit als Startquellen für den Random-Surfer fungieren.

Implementiert ist der PageRank-Algorithmus entsprechend Listing 3.3 in der Java-Klasse `PersonalizedPageRankAction` sowie in einer Version für beliebig gewichtete Zitationsgraphen in der Klasse `WeightedPersonalizedPageRankAction`. Die gewichtete Variante

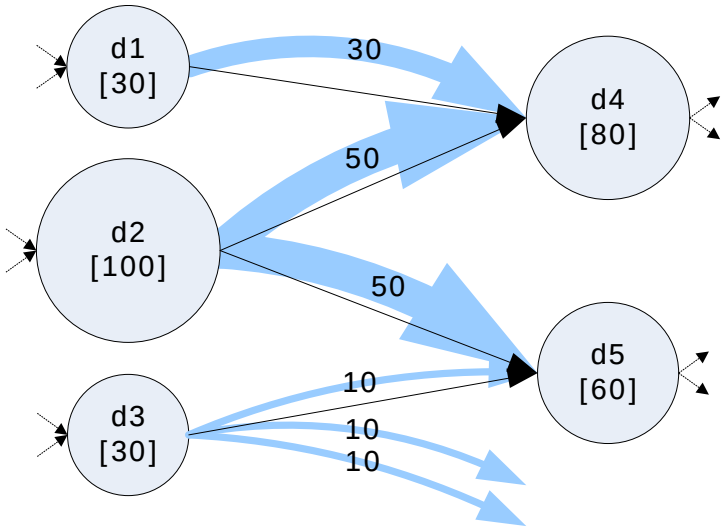


Abbildung 3.5.: Vereinfachte Berechnung der PageRank-Gewichte (ohne Normierung und Rank-Sources). Der PageRank wird anteilig an die referenzierten Knoten weiter gegeben.

Eingabe: Übergangsmatrix $\mathbf{A} \in \mathbb{R}^{n \times n}$,
Rank–Quellen $\mathbf{s} \in \mathbb{R}^n$
Ausgabe: PageRank $\mathbf{r} \in \mathbb{R}^n$

// Initialisierung
$\mathbf{r} \leftarrow \mathbf{s}$

wiederhole {
  // Neue PageRank–Gewichte bestimmen und normalisieren
  $\mathbf{r}' \leftarrow \mathbf{A} \cdot \mathbf{r}$
  $d \leftarrow \|\mathbf{r}\|_1 - \|\mathbf{r}'\|_1$
  $\mathbf{r}' \leftarrow \mathbf{r}' + d \cdot \mathbf{s}$
  // Verbesserung berechnen und stoppen, wenn gut genug
  $\delta \leftarrow \|\mathbf{r}' - \mathbf{r}\|_1$
  $\mathbf{r} \leftarrow \mathbf{r}'$
} **bis** $\delta \leq \epsilon$

Listing 3.3: Personalisierter PageRank

unterscheidet sich durch eine abgeänderte Übergangsmatrix:

$$A_{i,j} = \begin{cases} w_{i,j}/W_i, & \text{falls } i\text{-tes Dokument das } j\text{-te Dokument zitiert,} \\ 0, & \text{sonst.} \end{cases} \tag{3.16}$$

Hierbei bezeichnet W_i das Gesamtgewicht aller ausgehenden Kanten des i-ten Knoten.

Für einen azyklischen Zitationsgraphen kann man schon im Vorfeld einige Eigenschaften der resultierenden PageRank-Werte erwarten. Allein aufgrund der Graphenstruktur werden ältere Publikationen bevorzugt, wenn die Dokument immer nur den Referenzen einen Teilwert weitergeben. Für einen inversen Zitationsgraphen werden analog dazu neuere Dokumente bevorzugt. Daher könnte sich die PageRank-Bewertung evtl. dazu eignen Citation-Classics oder neuere Review-Artikel zu finden.

3.5.3. Hypertext Induced Topic Search

Hypertext Induced Topic Search (HITS) wurde 1997 von Jon Kleinberg vorgestellt (vgl. [Kle99]). Die Grundlage des HITS-Algorithmus bilden die sogenannten Hubs und Authorities. Dies sind zwei unterschiedliche Beurteilungen für Dokumente allein aufgrund ihrer Verlinkung. Es wird davon ausgegangen, dass gute Hubs Dokumente sind, welche auf möglichst viele gute Authorities verweisen. Im Gegenzug sind gute Authorities Dokumente, welche von vielen guten Hubs referenziert werden. Das Hub-Gewicht h_i und das Authority-Gewicht a_i des i-ten Dokuments berechnen sich wie folgt:

$$h_i = \lambda_1 \sum_{j=1}^{n} M_{ij} a_j \tag{3.17}$$

$$a_i = \lambda_2 \sum_{k=1}^{n} M_{ik}^T h_k \tag{3.18}$$

Das Hub-Gewicht eines Dokuments ist demnach die Summe aller Authority-Gewichte der referenzierten Dokumente. Umgekehrt ist das Authority-Gewicht die Summe aller Hubs-Gewichte der zitierenden Dokumente. Die Skalare λ_1 und λ_2 dienen lediglich zur Normierung der Hubs- und Authority-Gewichte. Graphisch veranschaulicht wird dieser Zusammenhang in Abbildung 3.6.

Die Gewichte lassen sich auch direkt als Vektoren über alle Dokumente beschreiben:

$$\mathbf{h} = \lambda_1 \mathbf{M}\mathbf{a} \tag{3.19}$$

$$\mathbf{a} = \lambda_2 \mathbf{M}^T \mathbf{h} \tag{3.20}$$

Die beiden Gleichungen ineinander eingesetzt ergeben

$$\mathbf{h} = \lambda_1 \lambda_2 \mathbf{M}\mathbf{M}^T \mathbf{h} \tag{3.21}$$

$$\mathbf{a} = \lambda_1 \lambda_2 \mathbf{M}^T \mathbf{M}\mathbf{a} \tag{3.22}$$

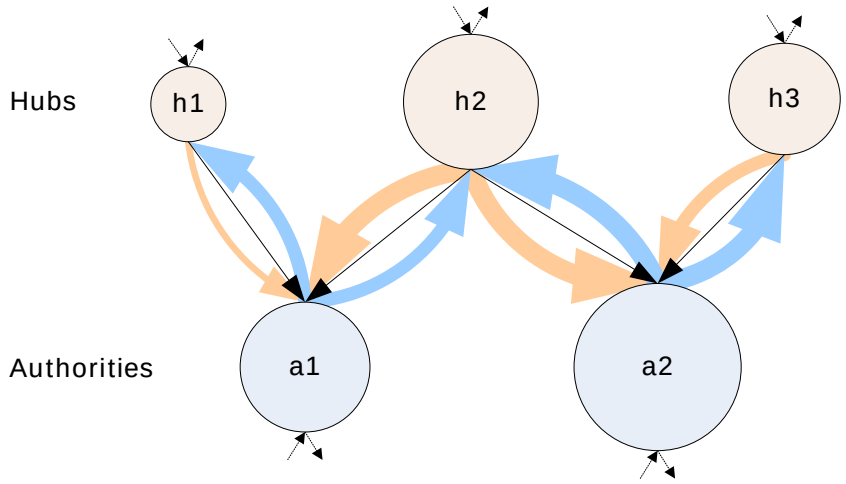


Abbildung 3.6.: Austausch der Hubs- und Authority-Gewichte zwischen miteinander verbundenen Knoten. Die Knotengröße repräsentiert entweder das Hub- oder das Authority-Gewicht.

Mögliche Lösungen $\mathbf{h}$ und $\mathbf{a}$ dieser Gleichungen stellen gerade die Eigenvektoren der Matrizen $\mathbf{M}\mathbf{M}^T$ und $\mathbf{M}^T\mathbf{M}$ dar. Diese beiden Matrizen sind bereits aus Abschnitt 3.2.4 als Koreferenz- und Kozitationsmatrizen bekannt. In [Kle99] wird zur Berechnung der jeweils größten Eigenvektoren der Algorithmus *Iterate* (siehe Listing 3.4) vorgeschlagen, welcher Ähnlichkeiten mit der Potenzmethode aufweist. In dieser Form fordert der Algorithmus keine festgelegte Genauigkeit, sondern führt eine durch k festgelegte Anzahl von Iterationen aus.

Implementiert ist HITS in den Java-Klassen `HitsRankAction` und `WeightedHitsRankAction`. Die erste Variante benutzt nur eine Verlinkungsmatrix und folgt damit den ursprünglichen Vorgaben aus [Kle99]. Die gewichtete Variante berücksichtigt auch die Kantengewichte in der Zitationsmatrix, was den Algorithmus aber selbst nicht ändert.

Das Hauptproblem von HITS ist die Beschränkung auf die Bestimmung des (betragmäßig) größten Eigenvektors. Dadurch wird nur das größte Themengebiet beachtet und auch nur für diese eine Dokumentenbewertung bestimmt. Selbst für unwesentlich kleinere Themengebiete werden deren Authorities keine große Bedeutung für die Gesamtbewertung zugeschrieben. Dadurch neigt HITS leicht dazu zu einem stärkeren Thema abzudriften. Auf Zitationsgraphen angewendet, zeigt sich beispielweise, dass solche starken Themengebiete eher theoretische und grundlegende Dokumente enthalten.

3.6. Polythematische Graphenanalyse mit PHITS

Wie wir gesehen haben, konzentriert sich das HITS-Verfahren auf das größte Themengebiet in einem Graphen indem es den ersten Eigenvektor bestimmt. Man kann nun durch eine Hauptkomponentenanalyse (PCA) noch weitere, kleinere Themenbereiche aus den

```
Eingabe: Zitationsmatrix M ∈ ℝ^{n×n}
Ausgabe: Hubs–Gewichte h ∈ ℝ^n,
         Authority–Gewichte a ∈ ℝ^n

// Initialisierung
h ← 1
a ← 1

wiederhole k mal {
  // Berechnung neuer Authority– & Hubs–Werte
  a' ← M^T h
  h' ← M a'
  // Gewichte normalisieren
  a ← a'/‖a'‖
  h ← h'/‖h'‖
}
```

Listing 3.4: Iterate-Algorithmus zur Bestimmung der Hubs- und Authority-Gewichte

Kozitation- und Koreferenzmatrizen bestimmen. Das Problem besteht darin, dass dabei ein unpassendes Statistikmodell benutzt wird, welches auch negative oder sehr große Zitierraten generieren kann (vgl. [CC00]).

Ein von statistischer Seite her passenderer Modell wird im PHITS-Verfahren benutzt, welches als probabilistisches Analogon von HITS in [CC00] vorgestellt wird. Es basiert auf dem allgemein gehaltenen Aspekt-Modell, welches hier als kurze Einführung in das Problemgebiets dient.

3.6.1. Aspekt-Modell

Das in [HP98] vorgestellte Aspekt-Modell ist ein Mischmodell für dyadische Daten. Bei dyadischen Daten beziehen sich gemachte Beobachtungen auf Paare (x, y) von Objekten aus zwei Mengen ($X = \{x_1, \ldots, x_I\}$ *und* $Y = \{y_1, \ldots, y_J\}$), also auf Elemente $(x, y) \in (X \times Y)$. Eine Beobachtung selbst ist im einfachsten Fall allein die Existenz eines Paars (x, y). Einer Beobachtung kann aber zusätzliche auch ein skalares Gewicht $w(x, y)$ zugeordnet werden.

Der Ausgangspunkt des Aspekt-Modells ist eine Sequenz von Beobachtungen $\mathcal{S} = \{(x^n, y^n) \in X \times Y\}_{n=1,\ldots,N}$ als eine Realisierung von N Paaren von Zufallsvariablen $(X^n, Y^n)_{n=1,\ldots,N}$. Nun wird jeder Beobachtung (x^n, y^n) eine versteckte Zufallsvariable A^n aus einer endliche Menge $\mathcal{A} = \{a_1, \ldots, a_K\}$ zugeordnet. Eine Realisierung $\vec{a} = (a^n)_{n=1,\ldots,N}$ partitioniert damit $\mathcal{S}$ in K Gruppen, welche auch Aspekte genannt werden.

Zur Entstehung der Beobachtungen werden zwei Annahmen gemacht:

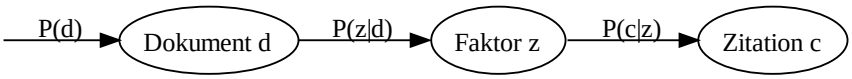


Abbildung 3.7.: Zitationsmodell in der asymmetrischen Parametrisierung

1. Die Paare $(x^n, y^n) \in \mathcal{S}$ sind gleichverteilt und unabhängig von anderen Paaren.
2. Die Zufallsvariablen X^n und Y^n sind unabhängig bei gegebenen Aspekt A^n.

Der zugrundeliegende Datengenerierungprozess und die weitere Modellerstellung wird im Folgendem direkt am PHITS-Verfahrens gezeigt. Eine tiefergehende allgemeine Beschreibung des Aspekt-Modells findet sich u. a. in [HP98] und [Ord05].

3.6.2. Zitationsmodell

Das PHITS-Verfahren basiert auf einem Zitationsmodell in dem eine Beobachtung aus jeweils einem Dokument und einer von diesem Dokument gemachten Zitierung besteht. Eine Beobachtungssequenz soll dann anhand einer kleinen Anzahl von versteckten Faktoren (in diesem Kontext auch zu verstehen als Themengebiete) erklärt werden. Das Zitationsmodell wird dazu in [CC00] als ein generativer Prozess beschrieben:

- Ein Dokument $d \in D$ wird mit der Wahrscheinlichkeit $P(d)$ erzeugt.
- Das Thema $z \in Z$ wird mit der Wahrscheinlichkeit $P(z|d)$ gewählt.
- Zitationen $c \in C$ werden mit der Wahrscheinlichkeit $P(c|z)$ erzeugt.

Dieser in Abbildung 3.7 veranschaulichte Generierungsprozess entspricht der asymmetrische Parametrisierung des Aspekt-Modells aus [HP98].

Für die Analyse von Zitationsnetzwerken sind die Dokumente und Zitationen normalerweise in der selben Sammlung, also $C = D$. Dessen ungeachtet haben sie im Zitationsmodell unterschiedliche Rollen, so dass für $c \in C$ und $d \in D$ mit $c = d$ in den folgenden Ausführungen nicht zu folgern ist, dass $P(c) = P(d)$ gilt. Eine Mitgliedschaft in C drückt hier eine Wird-Zitiert-Rolle aus und eine Mitgliedschaft in D eine Zitiert-Rolle.

Die Wahrscheinlichkeit $P(d, c)$ für eine Beobachtung von (d, c), also dass ein Dokument d das Dokument c zitiert, berechnet sich nach dem Zitationsmodell aus:

$$P(d, c) = P(d) \cdot P(c|d) \text{ mit} \tag{3.23}$$

$$P(c|d) = \sum_{z \in Z} P(c|z) \cdot P(z|d) \tag{3.24}$$

Die Gesamtwahrscheinlichkeit oder Likelihood P einer Menge von Beobachtungen S für ein gegebenes Modell θ ist demzufolge:

$$P(S|\theta) = \prod_{(d,c) \in S} P(d, c|\theta) \tag{3.25}$$

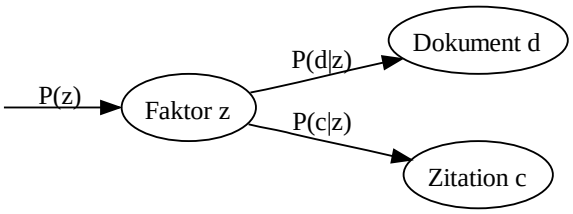


Abbildung 3.8.: Zitationsmodell in der symmetrischen Parametrisierung

Nach dem Satz von Bayes, welcher für zwei Ereignisse A und B lautet

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)} \tag{3.26}$$

lässt sich (3.23) auch umformen zu

$$P(d, c) = \sum_{z} P(z) \cdot P(c|z) \cdot P(d|z). \tag{3.27}$$

Dies entspricht der symmetrischen Parametrisierung (vgl. Abbildung 3.8) des Aspekt-Modells aus [HP98]. In dieser Form wird die im Aspekt-Modell geforderte Unabhängigkeit von $P(d|z)$ und $P(c|z)$ nochmal besonders deutlich.

Die Zitationen S werden nun am besten von dem Modell θ „erklärt“, welches die Gesamtwahrscheinlichkeit $P(S|\theta)$ maximiert:

$$\arg\max_{\theta} P(S|\theta) \tag{3.28}$$

3.6.3. Maximierung der Modellwahrscheinlichkeit

Zum Finden des wahrscheinlichsten Modells wird eine angepasste Version des EM-Algorithmus benutzt.

Statt das schwierig zu maximierende Produkts aus (3.25) zu maximieren wird die Log-Likelihood L maximiert:

$$L(S|\theta) = \sum_{(d,c)\in S} \ln P(d, c|\theta) \tag{3.29}$$

Die Log-Likelihood ist in der Regel einfacher zu maximieren als die Likelihood. Zudem kann es in der Praxis bei der Berechnung der Likelihood mit Fließkommazahl schnell zu einem Unterlauf kommen.

Der EM-Algorithmus besteht aus zwei Operationen, welche solange iteriert werden, bis es zu keiner weiteren nennenswerten Verbesserung mehr kommt:

Expectation Bestimmung der Erwartungswerte $P(z|d, c)$ als Schätzwert, dass ein Dokument-Zitations-Paar durch einen Faktor z erklärt wird

Maximization Anpassung der Modellparameter $P(z)$, $P(c|z)$ und $P(d|z)$ zur Maximierung der Likelihood $P(S|\theta)$ oder Log-Likelihood $L(S|\theta)$.

Der erste Schritt besteht somit aus der Berechnung von

$$P(z|d,c) = \frac{P(z) \cdot P(d|z) \cdot P(c|z)}{\sum_{z'} P(z') \cdot P(d|z') \cdot P(c|z')} \tag{3.30}$$

für jede mögliche Kombination von $z \in Z$ und $(d,c) \in E$.

Im zweiten Schritt werden die neuen Modellparameter bestimmt:

$$P(z) = \frac{\sum_{d,c} w(d,c) \cdot P(z|d,c)}{\sum_{d,c,z'} w(d,c) \cdot P(z'|d,c)} \tag{3.31}$$

$$P(d|z) = \frac{\sum_{c} w(d,c) \cdot P(z|d,c)}{\sum_{d',c} w(d,c) \cdot P(z|d',c)} \tag{3.32}$$

$$P(c|z) = \frac{\sum_{d} w(d,c) \cdot P(z|d,c)}{\sum_{d,c'} w(d,c) \cdot P(z|d,c')} \tag{3.33}$$

3.6.4. Initialisierung

Der Expectation-Schritt lässt sich nur mit bereits vorhandenen Werten aus dem zweiten Schritt ausführen. Für die erste Iteration sind diese noch nicht vorhanden, so dass anfangs entweder eine andere Schätzung oder auch eine zufällige Wahl der Parameter nötig ist.

3.6.5. Konvergenz

Der EM-Algorithmus konvergiert zu einem lokalen aber nicht zwingend globalen Optimum. Ein allgemeiner Lösungsansatz um ein Steckenbleiben in lokalen Maxima zu vermeiden ist das sogenannte *Simulated Annealing* (Simulierte Abkühlung). Es kann gezeigt werden, dass damit ein globales Maximum erreicht werden kann, sofern die Abkühlung langsam genug durchgeführt wird (vgl. [RN03, S.115]).

3.6.6. Generalisierung mit Tempered EM

Wie oben beschrieben ist es mit dem EM-Algorithmus möglich ein für eine Menge von Beobachtungen maximal angepasstes Modell zu erzeugen. In der Regel ist ein derartiges Modell aber zu speziell auf die zur Erstellung benutzten Beobachtungen angepasst. Dieses als *Overfitting* (Überanpassung) bezeichnete Problem hat zur Folge, dass das Modell unbekannte Beobachtungen nicht gut „erklären“ kann. Das bedeutet gleichzeitig, dass es offensichtlich die realen, eher einfachen Zusammenhänge nicht mehr korrekt beschreibt, sondern evtl. sehr spezielle Eigenschaften der Trainingsdaten. Auch für den Fall, dass

überhaupt keine unbekannten Beobachtungen existieren, wovon wir bei einem Zitationsgraph ausgehen, ist also Overfitting ein Problem, welches unter Umständen für einen Menschen völlig unerklärbare Ergebnisse liefert.

In [Hof99b, Hof99a] wird eine Methode zur Generalisierung des Maximum-Likelihood-Verfahrens für Mischmodelle namens *Tempered EM* (TEM) vorgeschlagen. Diese Methode ist eng verwandt mit dem *Deterministic Annealing* (siehe [Ros98]).

Für TEM ändert sich die Expectation-Berechnung (3.30) zu:

$$P(z|d,c) = \frac{[P(z) \cdot P(d|z) \cdot P(c|z)]^{\beta}}{\sum_{z'} [P(z') \cdot P(d|z') \cdot P(c|z')]^{\beta}} \tag{3.34}$$

Mit dem neu eingeführten Parameter β (als eine Art von inverser Temperatur) kann der Anteil der Entropie für die Erwartungswerte beeinflußt werden. Für $\beta = 1$ gibt es keinen Unterschied zu (3.30). Für $\beta < 1$ werden Erwartungwerte mit einer höheren Entropie bevorzugt und schließlich sind die Werte für $\beta = 0$ gleichverteilt. Anschaulich beschrieben bewirkt $\beta < 1$ also eine Glättung des Suchraums, sodass lokale Extrema abgeschwächt und vermieden werden können.

Normalerweise werden solche temperaturgesteuerten Verfahren, wie oben beschrieben, benutzt um lokale Maxima zu vermeiden. Dazu wird mit einer hohen Temperatur begonnen und die Temperatur bis zu ihrem Minimum verringert. Mit der ersetzten Berechnung (3.34) würde also bei einem kleinen $\beta < 1$ gestartet und suksessive bis auf $\beta = 1$ die Temperatur verringert. Im Gegensatz dazu wird in [Hof99b, Hof99a] vorgeschlagen, (3.34) zur Vermeidung von Overfitting einzusetzen.

Dazu wird eine einfache Methode vorgestellt um β mit Hilfe von Tuningdaten nach jeder Iteration anzupassen. Für die Anpassung wird ein Teil der Beobachtungen zurück gehalten und nach jeder Iteration die Likelihood dieser Tuning-Daten als Bewertungsgrundlage zur Generalisierung des Modells benutzt. Listing 3.5 zeigt mögliche Umsetzungen für dieses Verfahren. Dort wird der Tempering-Faktor β bis maximal zu einer unteren Schranke β_{min} verringert. Zusätzlich kann man unterschiedlich auf eine Verschlechterung der Likelihood für alle Daten reagieren. Diese Verschlechterung ist ein Anzeichen dafür, dass der Tempering-Faktor β zu klein wird. Zwei Fragen stellen sich: Ist es besser das neue, insgesamt schlechtere Modell weiter zu benutzen oder besser das vorige Modell zu behalten? Und sollte man besser die TEM-Schritte sofort beenden oder β bis β_{min} weiter verringern? Auch die Wahl der Parameter β_{min}, L_{limit} und dem Verringerungsfaktor η kann man nicht direkt angeben. Die Auswirkungen der verschiedenen Optionen und Parametern wird in Abschnitt 5.2.3 experimentell an Daten aus dem Cora-Datensatz untersucht.

3.6.7. Das Problem der unbeobachteten Rollen

Ein Problem bei Maximum-Likelihood-Lernverfahren besteht darin, dass für bestimmte (noch) nicht beobachtete Ereignisse in der gelernten Hypothese eine Wahrscheinlichkeit von 0 zugewiesen wird. Für ein nicht beobachtbar zitiertes Dokument c gilt bspw.

```
Eingabe: Trainingsdaten S_train und Tuningsdaten S_tune ,
         β_min , η , L_limit , finalIterations
Ausgabe: Wahrscheinlichkeitsmodell θ

// Initialisierung
θ ← RandomInit()
β ← 1.0

// Tempered EM – Schritte
solange β ≥ β_min wiederhole {

    θ' ← TEM(θ, S_train, β)

    // Keine Verbesserung für die Tuningsdaten?
    falls L(S_tune|θ') − L(S_tune|θ) ≤ L_limit tue {
        β ← η · β
    }

    // Verschlechterung für alle beobachtete Daten?
    falls L(S_train ∪ S_tune|θ') − L(S_train ∪ S_tune|θ) ≤ 0 tue {

        Option 1: // schlechteres Modell übernehmen
            θ ← θ'

        Option 2a: // early-stopping
            Schleife abbrechen

        Option 2b: // weiter machen
            falls β nicht verringert wurde tue {
                β ← η · β
            }

    } sonst {
        θ ← θ'
    }

}

// Finale Iterationen
tue finalIterations mal {
    θ ← TEM(θ, S_train ∪ S_tune, β_final)
}
```

Listing 3.5: Tempered-EM

$P(c) = 0$ und analog für ein nicht beobachtbar zitierendes Dokument d ist $P(d) = 0$. Auf den ersten Blick stellt dieses Verhalten für PHITS kein Problem dar, da das fertig gelernte Modell nicht dazu benutzt werden soll um auch unbekannte Beobachtungen zu erklären, sondern die bekannten Beobachtungen anhand unbekannter Faktoren. Das Problem zeigt sich, sobald man den vorgeschlagenen Algorithmus zur β-Anpassung benutzt. Dort wird ein Modell θ anhand einer Trainingsmenge gelernt und gleichzeitig versucht durch Anpassung von β die Likelihood der Tuningsmenge zu maximieren. Nun kann es passieren, dass in der Tuningsmenge eine Beobachtung (d, c) enthalten ist, für die das Modell $P(d, c|\theta) = 0$ liefert. Die Likelihood oder LogLikelihood der Tuningmenge wird daraufhin ebenfalls Null bzw. $-\infty$ und eine mögliche Verbesserung/Verschlechterung des Modells bzgl. der Tuningmenge kann daraus nicht mehr festgestellt werden.

Mehrere Möglichkeiten bieten sich an mit unbeobachteten Rollen in der Tuningmenge zu verfahren:

1. Die betreffende Beobachtung bei der Berechnung der Tuning-Likelihood wird einfach ignoriert. Damit bleibt die Tuning-Likelihood vergleichbar und die beobachtete Rolle wird nur in den evtl. folgenden finalen Iterationen im Modell berücksichtigt.
2. Bereits bei der Teilung der Beobachtungen in eine Trainings- und in eine Tuningsmenge darauf achten, dass unbeobachtete Rollen in der Tuningsmenge nicht vorkommen. Bei der Auswahl der Tuningmenge muss also darauf geachtet werden, dass suksessiv Beobachtungen so aus der Trainingsmenge genommen werden, dass die Rollen der beiden beteiligten Dokumente in der Trainingsmenge weiterhin repräsentiert bleiben.
3. Eine Minimalbeobachtung für alle möglichen $(d, c) \in D \times C$ in der Trainingsmenge. Hierfür ist eine vernünftige Implementierung nicht einfach. Das bloße Hinzufügen aller möglichen Minimalbeobachtungen ist für größere D und C aufgrund der sich stark erhöhenden Komplexität nicht praktikabel und eine Anpassung aller Berechnungsschritte ist relativ aufwendig.
4. Beim Verschieben in die Tuningsmenge ein wenig der einzelnen Beobachtungen in der Trainingsmenge belassen. Es verbleibt sozusagen eine Ahnung der Beobachtung in der Trainingsmenge. Dies kann derart geschehen, dass die betreffenden Beobachtungen aufgeteilt werden: Bspw. 1% des Beobachtungsgewichts verbleibt in der Trainingsmenge und die restlichen 99% kommen in die Beobachtung der Tuningsmenge.
5. Selbstzitierungen als Minimalbeobachtung. Jedes in der Dokumentenmenge enthaltene Dokument zitiert sich selbst mit einem kleinen, für alle Dokumente gleichbleibendes Gewicht. Dies ist wohl die einfachste Variante, welche auf einer Minimalbeobachtung basiert.

In dieser Arbeit wurde die zweite Lösung umgesetzt, da sie keine Informationen „verschenkt“ und eine Bewertung des Modells anhand völlig unbekannter Beobachtungen vornimmt. Bei allen anderen Lösungen haben die kritischen Beobachtungen in der Tuningsmenge weder einen Einfluß auf das zu lernende Modell noch helfen sie beim Tunen

des Algorithmus. Daher kann man solche Beobachtungen auch direkt in der Trainingsmenge belassen, womit sie zumindest bei der Modellberechnung einbezogen werden. Die dritte Möglichkeit kann aufgrund der praktischen Probleme direkt verworfen werden. Die letzten beiden Methode sind da vorzuziehen. Die vorletzte Methode hat den Nachteil, dass Beobachtungen in der Tuningmenge nicht völlig unabhängig zu den in der Trainingsmenge sind, sodass die Bewertung im Grunde nicht objektiv ist. Trotzdem ist dies eine einfache Alternative, welche sicherlich ebenfalls gute Ergebnisse liefert.

3.6.8. Dokumentenranking und weitere Statistiken

Die Parameter eines fertig gelernten Wahrscheinlickeitsmodells ermöglichen eine Vielzahl von Auswertmöglichkeiten der darin beschriebenen Dokumente und Faktoren.

Für ein Dokumentenranking bzgl. eines Faktors oder Themas können die bedingten Wahrscheinlichkeiten $P(c|z)$ als Analogon zu einer Authority-Bewertung sowie $P(d|z)$ als Analogon zu den Hubs-Gewichten benutzt werden. Diese beiden Werte sind direkt im symmetrischen Zitationsmodell angegeben, sodass eine Umrechnung der Wahrscheinlichkeiten nicht mehr notwendig ist.

Die Modellparameter können aber auch genutzt werden um Dokumente zu klassifizieren. Die bedingte Wahrscheinlichkeit

$$P(z|c) = \frac{P(c|z)P(z)}{P(c)} = \frac{P(c|z)P(z)}{\sum_{z'} P(c|z')} \tag{3.35}$$

kann dazu benutzt werden die hauptsächlichen Faktoren (resp. Themen) zu ermitteln, welche das untersuchte Dokument c zitieren. Analog kann die bedingte Wahrscheinlichkeit $P(z|d)$ zu einer Klassifizierung eines Dokuments d auf Basis der zitierten Themen benutzt werden.

4. Citana

Die in den vorigen Kapiteln vorgestellten Methoden wurden in der Java-Applikation „Citana“ implementiert. Da die Anforderungen an das Programm anfangs recht vage waren, wurde Citana nach dem Konzept des Evolutionären Prototyping entwickelt. Dabei wurde besonders darauf geachtet, die Programmstruktur möglichst einfach und allgemein zu halten. Der überwiegende Anteil der benutzten Klassen basieren auf Interfaces und abstrakten Klassen. In den folgenden Abschnitten werden allgemeine Informationen zur Entwicklungsumgebung sowie die wichtigsten Interfaces und Klassen mit ihren Methoden und Beziehungen vorgestellt.

Dieses Kapitel ist in folgende Abschnitte unterteilt: Der erste Abschnitt „Zitationsanalyse mit Citana“ gibt es eine Einführung zur Benutzung der verschiedenen Funktionen von Citana. Die letzten beiden Abschnitte zeigen den grundlegenden Aufbau der Programmimplementierung. Der erste Teil davon beschreibt die wichtigsten bibliographischen Datenstrukturen und anschließen wird der grundlegende Aufbau der graphischen Benutzeroberfläche betrachtet.

4.1. Zitationsanalyse mit Citana

Das Hauptprogramm Citana wurde als Applikation mit graphischer Benutzeroberfläche entwickelt und bietet keine nennenswerten Anwendungsmöglichkeiten über einer Textkonsole.

Zum Starten von Citana kann die Batch-Datei `Citana.bat` aufgerufen werden. Darin kann auch der dem Programm zur Verfügung gestellte Speicher angepasst werden.

Abbildung 4.1 zeigt das Hauptfenster von Citana. In der Symbolleiste ① sind die häufiger benutzten Aktionen für die aktuelle Ansicht direkt verfügbar. In der Baumansicht ② werden alle verfügbaren Ansichten angezeigt, welche im Ansichtsbereich ③ verwendet werden können. Das Textfeld ④ dient zur Ausgabe von diversen Informationen des laufenden Programms.

4.1.1. Ansichten und ihre Funktionen

Wie schon erwähnt sind in der Baumansicht die verfügbaren Ansichten einsortiert. Eine einfache Ansicht besteht z. B. nur aus einer Dokumentensammlung. Darauf können dann weitere Ansichten wie verschiedene Rangordnungen oder Gruppierungen aufbauen. Über

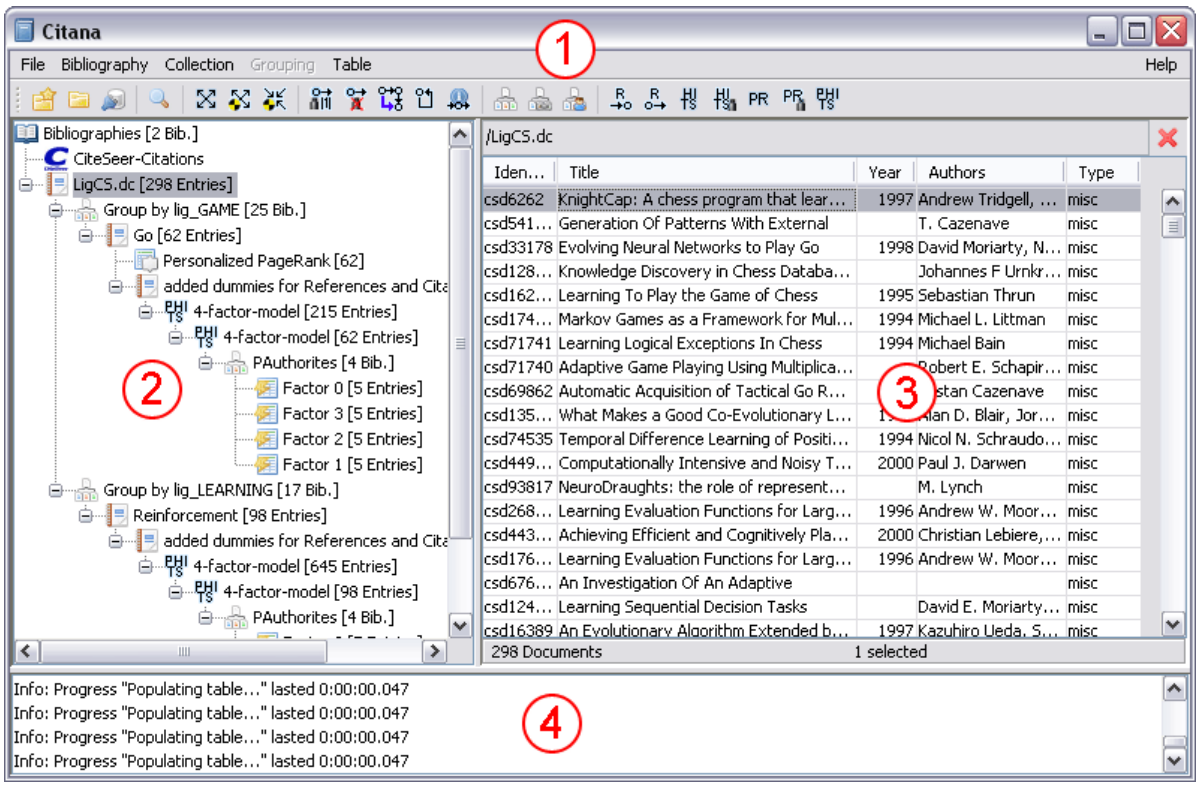


Abbildung 4.1.: Hauptfenster von Citana

diesen Baum kann nicht nur die aktuelle Ansicht gewählt werden, sondern jeder Baumknoten besitzt auch ein eigenes Kontextmenü, welches wichtige Funktionen enthält. Hier sind für spezielle Knoten (z. B. bei PHITS-Knoten) wichtige Aktionen enthalten, die ansonsten nicht über das Hauptmenü oder die Symbolleiste verfügbar sind.

Dargestellt wird der überwiegende Anteil der Daten in Tabellenform. Hierzu muss man wissen, dass normalerweise nicht alle Spalten sofort angezeigt werden. Über die Kopfzeile der Tabelle ist ein weiteres Kontextmenü verfügbar um Spalten ein- und auszublenden. Durch normales Anklicken der Spaltenbezeichner können die Zeilen der ausgewählten Spalte entsprechend sortiert werden, was insbesondere für die verschieden Ranglisten wichtig ist.

4.1.2. Importieren von Bibliographien

Es gibt drei unterschiedliche Möglichkeiten bibliographische Daten in Citana zu importieren. Die erste Form ist das Laden von Dokumentensammlungen aus Dateien. Mögliche Formate sind BibTeX, BibTeXML und das programmeigene an BibTeXML angelehnte Format mit Endung „.dc“. Der BibTeX-Import ist nicht sehr robust implementiert und lädt noch nicht alle BibTex-Dateien korrekt. Deshalb wird für BibTeX-Daten eine vorhe-

rige Konvertierung nach BibTeXML (z. B. mit Jabref [5]) empfohlen. Zitationsinformationen können aber nur im eigenen DC-Format enthalten sein; BibTeX und BibTeXML stellen in ihren Standards dafür keine Datenfelder bereit.

Die zweite Möglichkeit mit externen Daten zu arbeiten ist eine Verbindung zu einer MyCiteSeer-Datenbanken aufzubauen. Die gewählte Bezeichnung „MyCiteSeer“ soll die Verbindung von MySQL und einer Zitationsdatenbank ausdrücken. Diese Datenbank muß aber nicht zwangsläufig CiteSeer darstellen. Die Verbindung zu einer MyCiteSeer-Datenbank ermöglicht es mit sehr großen Bibliographien zu arbeiten, welche möglicherweise nicht in den verfügbaren Hauptspeicher passen. Die Struktur einer MyCiteSeer-Datenbank wird in Anhang C beschrieben. Dort werden auch die vorhandenen Hilfsprogramme zum Import von CiteSeer-, CORA- und DBLP-Datensätze kurz vorgestellt.

Die letzte Form von externen Daten gibt es in „CiteSeer-Citation“. Diese automatisch geöffnete Bibliographie bezieht ihre Daten direkt über das Web-Interface von CiteSeer. In dieser Bibliographie kann auch auf die nur als Zitationen vorhandenen Publikationen zugegriffen werden.

4.1.3. Dokumentensammlungen und deren Ansicht

Mit der Ausnahme von „CiteSeer-Citation“ sind alle Bibliographien in Citana Dokumentensammlungen. Eine Dokumentensammlung zeichnet sich dadurch aus, dass alle enthaltenen Dokumente mit ihren Eigenschaften aufgelistet werden können. „CiteSeer-Citation“ repräsentiert keine Dokumentensammlung, da es nicht möglich ist die in CiteSeer enthaltenen Dokumente oder Zitationen über das Webinterface direkt zu erhalten. Mit allen Dokumentensammlungen kann man grundsätzlich gleich verfahren, wobei aber für große Sammlungen viele Methoden nicht ausgelegt sind und evtl. so viel Speicher oder Zeit benötigen, dass sie effektiv nicht benutzt werden können. Für solche Fälle kommt man evtl. mit den in Anhang C.3 beschriebenen Verfahren einfacher zu einem Ergebnis.

Eine wichtige Rolle für die Arbeit mit Bibliographien in Citana spielen ihre Positionen in der Baumansicht. Die oberste Baumebene grenzt in gewisser Weise die Namensräume der verschiedenen Bibliographien ab. Alle unter einer Hauptbibliographie liegenden Dokumentensammlungen dürfen daher nur Dokumente aus einer solchen Hauptbibliographie selbst oder äquivalente Dokumente mit gleichen Bezeichnern enthalten. Im Allgemeinen liefern auch nur Bibliographien der obersten Baumebene Referenzinformationen für die Dokumente der darunter liegenden Sammlungen. Die Graphenstrukturen können aber durchaus auch geändert werden und z. B. eine andere Linkgewichtung enthalten. Unterbibliographien sind immer das Ergebnisse einer Funktion wie z. B. Suchen oder Expandieren.

Eine Reihe von statistischen Angaben über eine Dokumentensammlung läßt sich über den Informationen-Menüpunkt ermitteln. Ermittelt werden u. a. die Anzahl von Dokumenten, welche nicht zitiert werden, Dokumente welche selbst nicht zitieren und Dokumente zu denen keinerlei Referenzinformationen existieren.

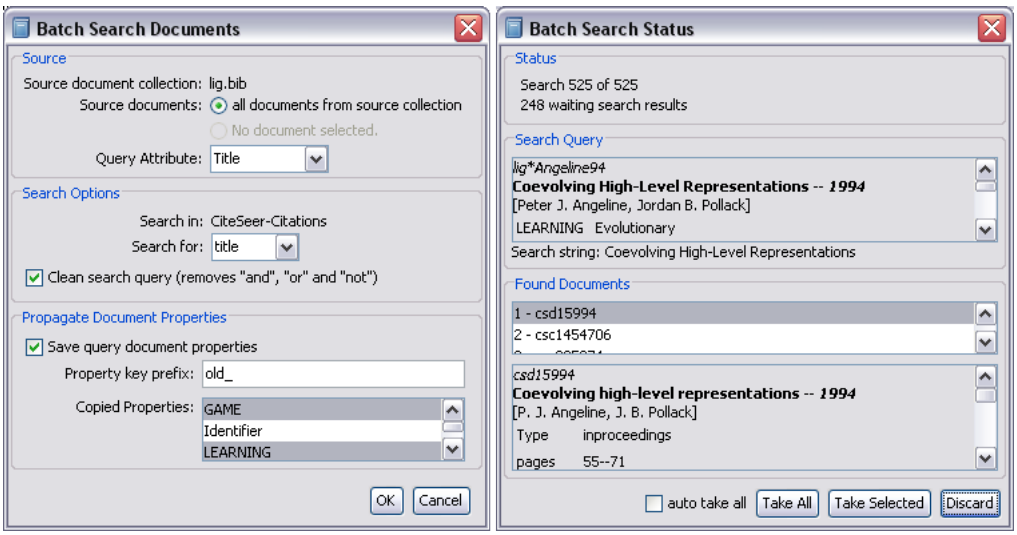


Abbildung 4.2.: Links: Suchoptionen für die gleichzeitige Suche von mehreren Dokumenten in einer Bibliographie. Rechts: Manuelle Überprüfung der Suchergebnisse

4.1.4. Suchen, Matchen und Transferieren von Dokumentensammlungen

Unterschiedliche Bibliographien oder Zitationsdatenbanken haben in der Regel ihre eigenen Dokumentenbezeichner. Gerade für die Arbeit mit Literaturlisten ohne Zitationsdaten ist es wichtig zu den Dokumenten entsprechende Einträge in einer anderen Bibliographie mit eben diesen Informationen zu finden. Zu diesem Zweck gibt es Funktionen zur Suche und zum Matchen von ganzen Dokumentensammlungen. Diese beiden Funktionen folgen zwei unterschiedlichen Ansätzen. Bei einer Dokumentensuche besteht die Anfrage aus einem einzelnen String, z. B. dem Dokumententitel. Beim Matchen besteht die Anfrage aus ganzen Dokumenten zu denen andere passende Dokumente gefunden werden sollen. Dies entspricht der Duplikaterkennung aus Abschnitt 3.1.

Eingeleitet werden können diese Funktionen durch Drag und Drop der Quellsammlung auf die Zielbibliographie, woraufhin ein Popup-Menü die genaue Operation abfragt. Möglich ist hier auch nur einzeln selektierte Dokumente aus einer Sammlung auf den Zielknoten zu ziehen. Handelt sich das Ziel um eine Dokumentensammlung, kann die Quellsammlung darin gesucht, gematcht oder transferiert werden. Falls sie keine Dokumentensammlung darstellt, fällt die Option des Matchens weg.

Für die Suche muß im zugehörigen Dialog (vgl. Abbildung 4.2 links) die Dokumenteneigenschaft, welche als Suchanfrage benutzt werden soll sowie die Art der Suche gewählt werden. Dabei sind die wählbaren Dokumenteneigenschaften und die Suchart abhängig von den jeweiligen Bibliographien. Nur nach einem oder mehreren (mit Komma getrennten) Dokumentenbezeichnern kann standardmäßig in jeder Bibliographie gesucht werden. Im Weiteren kann die Suchanfrage „gesäubert“ werden. Dadurch werden die

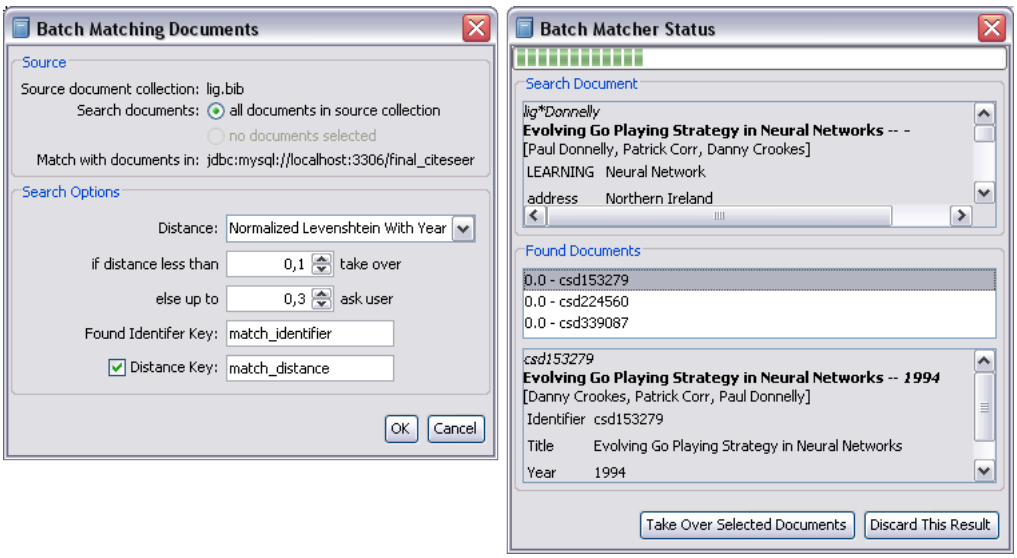


Abbildung 4.3.: Links: Optionen für das Matchen von Dokumenten aus zwei Sammlungen. Rechts: Manuelle Überprüfung von fraglichen Ergebnissen

Wörter „and“, „or“ und „not“ entfernt, welche (abhängig von der verwendeten Suchart) Suchoperatoren sein können. Als letzter Punkt können einzelne Eigenschaften des Anfragedokuments zu den Eigenschaften der gefundenen Dokumente übertragen werden. Die Ergebnisse der Suchanfragen werden zur manuellen Überprüfung in einem weiteren Dialog (vgl. Abbildung 4.2 rechts) gesammelt. Zu jedem Suchdokument werden die gefundenen Dokumente mit ihren Eigenschaften angezeigt und können dann als „korrekte“ Funde übernommen werden.

Für das Matchen von Dokumenten (vgl. Abbildung 4.3 links) kann neben der Distanzfunktion angegeben werden, inwieweit die Ergebnisse automatisch übernommen werden sollen. Dazu wird ein Intervall für die Distanzen angegeben, welche manuell überprüft werden sollen. Alle gefundenen Ergebnisse mit größerer Distanz werden automatisch verworfen und alle Funde mit kleinerer Distanz automatisch übernommen. Bei mehr als einem Dokument unterhalb der Übernahmegrenze wird jedoch immer eine manuelle Bestätigung verlangt (vgl. Abbildung 4.3 rechts). Im Gegensatz zur oben beschriebenen Suche ist das Ergebnis beim Matchen keine Dokumentensammlung unterhalb der zu durchsuchenden Sammlung, sondern eine Sammlung der Anfragedokumente mit einer neuen Eigenschaft, welche die Bezeichner der passenden Dokumente enthält. Optional können auch die zugehörigen Dokumentdistanzen als eine neue Eigenschaft gespeichert werden. Um die Sammlung schließlich in der verglichenen Bibliographie benutzen zu können, reicht eine einfache Bezeichner-Suche mit der neuen Eigenschaft als Suchanfrage.

Bei der Arbeit mit verschiedenen Dokumentensammlungen tritt meist noch ein Problem auf, das theoretisch auch mit der Suchfunktion gelöst werden kann. Für den Fall, dass man eine vorher gespeicherte Teilsammlung einer Bibliographie wieder öffnet, wird

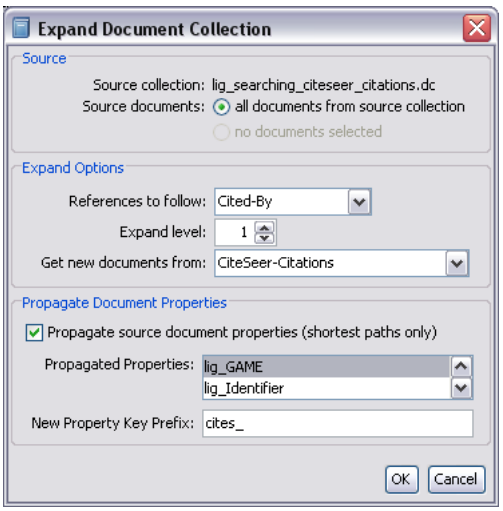


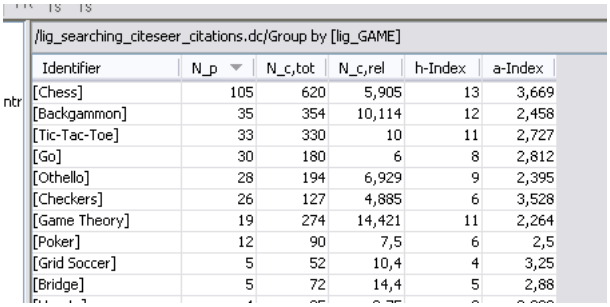
Abbildung 4.4.: Optionen zur Erweiterung einer Dokumentensammlung

diese Sammlung als eigene Hauptbibliographie in den Bibliographiebaum eingefügt. Dadurch wird verhindert, dass Dokumente mit evtl. fremden Bezeichnern ungeprüft benutzt werden. Es gibt damit aber auch keine Möglichkeit den Graph zu erweitern und es existieren auch nur die evtl. mitgespeicherten Zitationsinformationen. Um eine Dokumentensammlung direkt in eine fremde Hauptbibliographie zu übernehmen, gibt es die Funktion „Transferieren“. Damit werden alle Zitationsinformationen der neuen Elternbibliographie benutzt und die eigenen verworfen. Sinnvoll ist diese Funktion aber nur, wenn beide Bibliographien bereits für die gleichen Dokumente die gleichen Bezeichner benutzen. Eine Überprüfung, ob die Dokumente überhaupt in der neuen Elternbibliographie existieren, findet nicht statt.

4.1.5. Expandieren von Sammlungen

Um weitere, einer Sammlung möglicherweise thematisch zugehöriger Dokumente zu finden, ist es notwendig die Sammlung zu erweitern. Da keine Volltexte vorliegen, welche verglichen werden können, können dafür nur die Beziehungen der Dokumente untereinander genutzt werden. In Citana gibt es die Möglichkeit eine Dokumentensammlung anhand verschiedener Beziehungen und auf Basis einer Quellbibliothek zu expandieren. Abbildung 4.4 zeigt die vorhandenen Einstellungen für diese Funktion. Grundlegend ist die Wahl der Beziehung, welche zur Erweiterung benutzt werden soll. Derzeit mögliche Arten sind: „zitiert“, „wird-zitiert-von“, „zitiert oder wird-zitiert-von“, „kozitiert“ und „koreferenziert“. Die nächste Einstellung ist die Tiefe der Erweiterung. Normalerweise sollte eine Erweiterung um eine Ebene für die meisten Fälle ausreichen. Wichtig ist noch eine Quellbibliographie anzugeben, von der die hinzuzufügenden Dokumente entnommen werden können.

Zusätzlich zur reinen Erweiterung können den neuen Dokumenten Eigenschaften auf



/lig_searching_citeseer_citations.dc/Group by [lig_GAME]

Identifier	N_p	N_c,tot	N_c,rel	h-Index	a-Index
[Chess]	105	620	5,905	13	3,669
[Backgammon]	35	354	10,114	12	2,458
[Tic-Tac-Toe]	33	330	10	11	2,727
[Go]	30	180	6	8	2,812
[Othello]	28	194	6,929	9	2,395
[Checkers]	26	127	4,885	6	3,528
[Game Theory]	19	274	14,421	11	2,264
[Poker]	12	90	7,5	6	2,5
[Grid Soccer]	5	52	10,4	4	3,25
[Bridge]	5	72	14,4	5	2,88

Abbildung 4.5.: Beispiel für eine Gruppierung nach der (mehrwertigen) Eigenschaft „lig_GAME“ sortiert nach der entstehenden Sammlungsgröße

Grundlage ihrer Ursprungsdokumenten zugefügt werden. Ein Beispiel soll das verdeutlichen: Die Ursprungsammlung besteht aus Dokumenten, welche die Eigenschaft „Kategorie“ in verschiedenen Ausprägungen besitzen. Nun wird diese Sammlung erweitert indem alle zitierenden Dokumente hinzugefügt werden. Den neu hinzugefügten Dokumenten kann nun die Eigenschaft „zitiert_Kategorie“ mit den entsprechenden Werten zugewiesen werden. Wenn ein neues Dokument mehrere Dokumente unterschiedlicher Kategorien der Ursprungssammlung zitiert, werden alle Kategorien (mit Komma getrennt) gespeichert. Wird um mehr als eine Ebene erweitert, werden nur die Kategorien der Ursprungsdokumente gespeichert, welche den kürzesten Pfad besitzen.

Neben der Erweiterung durch „real“ vorhandene Dokumente einer übergeordneten Bibliographie existiert auch die Möglichkeit Dummy-Dokumente anzulegen, welche die gleichen Beziehungen des Originals haben, soweit diese bekannt sind. Diese können dann in diversen Analysen mitbetrachtet und anschließend aus den Ergebnissen wieder entfernt werden.

4.1.6. Gruppieren von Dokumenten

Eine Dokumentensammlungen kann nach unterschiedlichen Kriterien unterteilt werden. Die Dokumente können aufgrund ihrer Graphenstruktur oder basierend auf Eigenschaftswerte gruppiert werden. Bei der Gruppierung nach der Graphenstruktur ist es möglich Zusammenhangskomponenten von Zitationsgraphen zu finden. Darüber hinaus ist es möglich auch Zusammenhangskomponenten von Kozitations- oder Koreferenzgraphen zu bestimmen. Zusätzlich kann ein Mindestgewicht für die Links angegeben werden, welches gelten muss, damit zwei Dokumente als verbunden gelten. Zusammenhangskomponenten, welche lediglich aus einem Knoten bestehen, werden in einer eigenen Sammlung zusammengefasst.

Bei der zweiten Art der Gruppierung werden Dokumente nach gleichen Werten bestimmter Eigenschaften zusammengefasst. Ein einfaches Beispiel ist die Gruppierung nach Jahreszahlen. Möglich ist auch eine Gruppierung nach mehreren Eigenschaften gleichzeitig,

z. B. nach Jahr und Journal. Einige Eigenschaften können auch mehrere Werte gleichzeitig enthalten, wie z. B. Autoren oder eine bestimmte Form von Kategorisierung. Optional können solche Eigenschaftswerte geteilt und einzeln betrachtet werden, wodurch ein Dokument auch in mehreren Gruppen enthalten sein kann. Dadurch können sich die einzelnen Gruppen auch überlappen. Die Namen der einzelnen Gruppen (als eigenständige Dokumentensammlungen) bestehen aus den zusammengefassten Eigenschaftswerten.

4.1.7. Ranking von Dokumentensammlungen

Eine Gruppierungen wird normalerweise wie in Abbildung 4.5 als Tabelle dargestellt. Jede Zeile steht für eine Gruppe in Form einer Dokumentensammlung. In dieser Tabelle lassen sich weitere Spalten mit Informationen, wie absolute/relative Zitierhäufigkeit oder h-Index, einblenden.

Im Weiteren kann eine Häufigkeitsmatrix über Dokumenteneigenschaften erstellt werden.

4.1.8. Ranking von Dokumenten

Zur Zeit sind alle Ranking-Methoden aus Abschnitt 3.5 implementiert: InRank bewertet Dokumente nach Anzahl und Gewichte der Zitationen aus der Bibliographie. OutRank bewertet Dokumente nach der Anzahl und Gewichte von in der Bibliographie zitierten Dokumenten. Im Weiteren gibt es noch HITS, welches Dokument nach Hubs- und nach Authorities-Gewichten bewertet, PageRank bewertet Dokumente anhand einem Wahrscheinlichkeitswert und PHITS, welches Dokumente auf Grundlage der Wahrscheinlichkeiten eines Zitationsmodell bewertet.

Eine Ranking ist mit einer Ausnahme nur für eine einzige Ansicht gültig. Es ist z. B. nicht möglich zuerst PageRank und anschließend HITS auszuführen und beide Ranking zusammen anzeigen zu lassen. Die Ausnahme stellt die Funktion zur Dummy-Entfernung dar. Es ist also möglich aus einem Ranking oder einem PHITS-Modell alle Dummy-Dokumente zu entfernen und sich die Ergebnisse ohne sie anzeigen zu lassen.

4.1.9. Daten exportieren

Da Citana nur wenige Funktionen zur Visualisierung der berechneten Daten besitzt, gibt es Funktionen um die Daten möglichst universell weiterverarbeiten zu können. Komplette Dokumentensammlungen mit Referenzinformationen können nur im eigenen DC-Format gespeichert werden. Falls man nur an der Referenzstruktur interessiert ist, kann eine Sammlung im DOT-Format exportiert werden. DOT ist eine Auszeichnungssprache des Programmpakets Graphviz [4] zur Visualisierung von Graphen.

Alle weiteren Daten werden in der Regel in Tabellen dargestellt und können in einheitlicher Weise exportiert werden. Die Tabellenansichten können komplett als CSV-Datei gespeichert und komplett oder nur einzelne ausgewählte Zeilen in die Zwischenablage

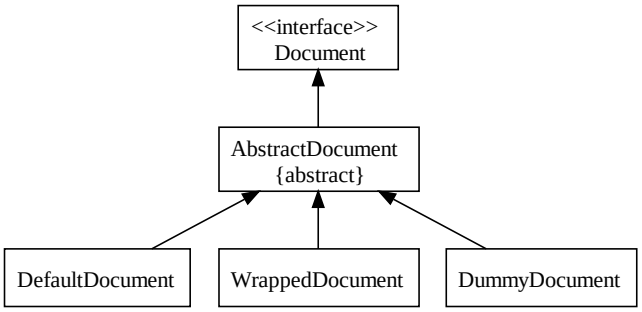


Abbildung 4.6.: Klassendiagramm für die öffentlichen `Document`-Klassen

kopiert werden. Bei allen drei Möglichkeiten werden ausschließlich die sichtbaren Spalten exportiert. Die Spaltenbezeichner selbst werden dabei nur für komplette Tabellen mitexportiert. Besonders der Austausch über die Zwischenablage macht es einfach die Daten bspw. in einem Tabellenkalkulationsprogramm zu sammeln und zu visualisieren.

4.2. Bibliographische Datenstrukturen

Die wichtigsten Schnittstellen zur Verwaltungen der bibliographischen Daten sind die Interfaces `Document`, `LinkManager`, `Bibliography` mit der davon abgeleiteten `DocumentCollection`, `RankModel` und `Grouping`.

4.2.1. Document

Ein Objekt der Klasse `Document` beschreibt eine Publikation durch einen eindeutigen Bezeichner und den bibliographischen Angaben, wie Titel, Publikationsjahr, Autoren, Dokumententyp und weiteren. Die Dokumenten-Bibliographien-Beziehung basiert auf unveränderbaren Dokumentenobjekten. Das bedeutet, dass ein `Document`-Objekt seine Eigenschaften nach seiner Instanzierung nicht mehr ändern darf (vergleichbar mit String-Objekten in Java). Dies hat den Vorteil, dass `Document`-Objekte gleichzeitig in verschiedenen Bibliographien enthalten sein können ohne auf möglichen Änderungen der `Document`-Objekte reagieren zu müssen. Es ist aber (im Gegensatz zu String) möglich auch eigene `Document`-Klassen zu implementieren.

Abbildung 4.6 zeigt die Klassenhierachie der öffentlichen `Document`-Klassen. Die Standardimplementierung ist `DefaultDocument`. Daneben gibt es Wrapper-Dokumente, welche eine existierende `Document`-Instanz um zusätzliche Eigenschaften erweitert. Schließlich gibt es noch Dummy-Dokumente, welche nur als Platzhalter für ein Dokument zu verstehen sind und in der Regel bis auf einen Bezeichner und einem generischen Titel keine Eigenschaften besitzen. Zur besseren Unterscheidung zu normalen Dokumenten, liefert die Dokumententyp-Eigenschaft solcher Dokumente den Wert „dummy“.

4.2.2. LinkManger

Ein `LinkManger` verwaltet Beziehungen zwischen, mittels eindeutigen Bezeichnern identifizierbaren, Objekten. Im Kontext einer Graphenstruktur enthält ein `LinkManger` also die (gewichteten) Kanten des Graphs. Die Links oder Kanten werden dabei immer als gerichtet angesehen. Zu einem Objekt können grundsätzlich die ausgehenden sowie die eingehenden Kanten sowie ihre Gewichte abgefragt werden. Die von `AbstractUndirectedLinkManger` abgeleiteten Klassen (vgl. Abbildung 4.7) sind nur insoweit ungerichet, dass eine Abfrage der ausgehenden Kante eines Objekts das gleiche Ergebnis liefert wie die nach eingehenden Kanten.

Die derzeitig vorhandenen Link-Manager (vgl. Abbildung 4.7) sind:

DefaultLinkManger ist die Standardimplementierung, welche Methoden zum Hinzufügen und Löschen von Links bietet. Gespeichert werden die Beziehungsinformationen in zwei Hash-Maps für einen direkten Zugriff auf ausgehende sowie eingehende Kanten.

InvertedLinkManger dreht die Richtung der Links eines existierenden Managers um. Aus eingehenden Links werden ausgehende und vice versa.

Cached[Undirected]LinkManger bietet für bereits existierende Link-Manager eine Caching-Funktion. Die Nutzung eines solchen Caches macht besonders für die LinkManager Sinn, die größere Mengenoperationen für jede Referenzabfrage durchführen müssen. Bei gerichteten Kanten werden beide Richtungen getrennt gecached, während die ungerichtete Version dahingehend keine Unterscheidung machen muss.

UndirectedLinkManger erstellt aus einem gerichteten LinkManager einen ungerichteten, indem er zu jeder ausgehenden Kante eine eingehende Kante hinzufügt und umgekehrt.

CoCitationManager liefert zu gegebenen Zitationsrelationen die zugehörigen (ungerichteten) Kozitations-Links, wie in Abschnitt 3.5 beschrieben. Gewichtet werden die Kanten durch die Vielfachheit der Kozitation.

CoReferenceManager liefert zu gegebenen Zitationrelationen die zugehörigen (ungerichteten) Koreferenzen, wie in Abschnitt 3.5 beschrieben. Gewichtet wird diese durch die Anzahl der gemeinsam zitierten Dokumenten.

HyperLinkManger liefert eine Verknüpfung über zwei Link-Manager, welche der Gruppenbeziehung aus Abschnitt 3.5 entspricht. Der erste LinkManger gibt dazu die Relation zwischen Sammlungen und deren Dokumenten an und der zweite Manager enthält die normalen Zitationsbeziehungen zwischen den Dokumenten.

Zusätzlich gibt es noch vier, eher experimentelle `LinkManager`:

NormalizedOutLinkManager normalisiert die Linkgewichte eines gegebenen LinkManager derart, dass die Summe der ausgehenden Linkgewichte eines Objekts Eins ergibt. Vorher vorhandene Linkgewichte werden dabei nicht beachtet.

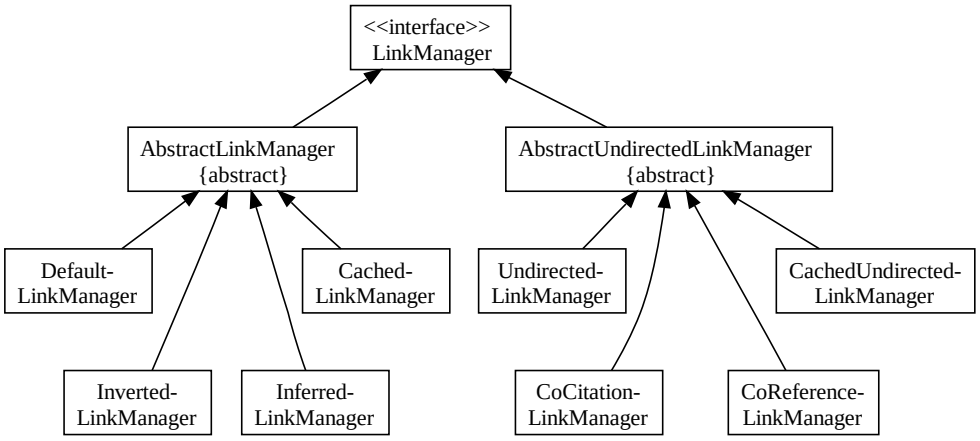


Abbildung 4.7.: Klassendiagramm für die wichtigsten `LinkManager`-Klassen

UnweightedLinkManager setzt alle Linkgewichte auf den Wert Eins; vorige Gewichte werden damit entfernt.

OneHopLinkManager fügt zusätzlich Links zu einem bestehenden LinkManager hinzu, welche einen Knoten überspringen. Genauer werden für alle Wege im Graph mit der Länge 2, explizite Links hinzugefügt, dessen Gewicht das Produkt der beiden Kanten beträgt.

SelfLoopManager fügt Schleifen, also Kanten mit gleichem Start- und Zielknoten, mit einem beliebigen Gewicht hinzu.

4.2.3. Bibliography und DocumentCollection

Objekte, welche das Interface `Bibliography` implementieren beschreiben Bibliographien in denen nach Dokumenten gesucht werden kann und passende Zitationsbeziehungen liefert. Das Interface `DocumentCollection` ist eine Konkretisierung von `Bibliography`. Implementierende Objekte stellen zusätzlich Dokumentensammlungen mit einer bekannten Größe und iterierbaren Einträgen dar.

Die Beziehungen der Dokumente untereinander, also die „zitiert“- und „wird-zitiert-von“-Relationen, werden von einem `LinkManager` in der `Bibliography` verwaltet. Zu einem Dokument liefert dieser Klasse die zugehörigen Referenzen oder Zitationen. Die zentrale Verwaltung der Referenzen hat gegenüber der Speicherung in den (unveränderlichen) `Document`-Objekten die Vorteile, dass die Konsistenz der entstehenden Referenzpaare (eine einzelne Beziehung muss für beide Richtungen indiziert werden) einfacher zu gewährleisten ist und dass unterschiedliche Beziehungsstrukturen allein durch den Wechsel des Link-Managers gebildet werden können.

Die derzeitig implementierten Bibliographien und Dokumentenkollektionen (vgl. Abbildung 4.8) sind:

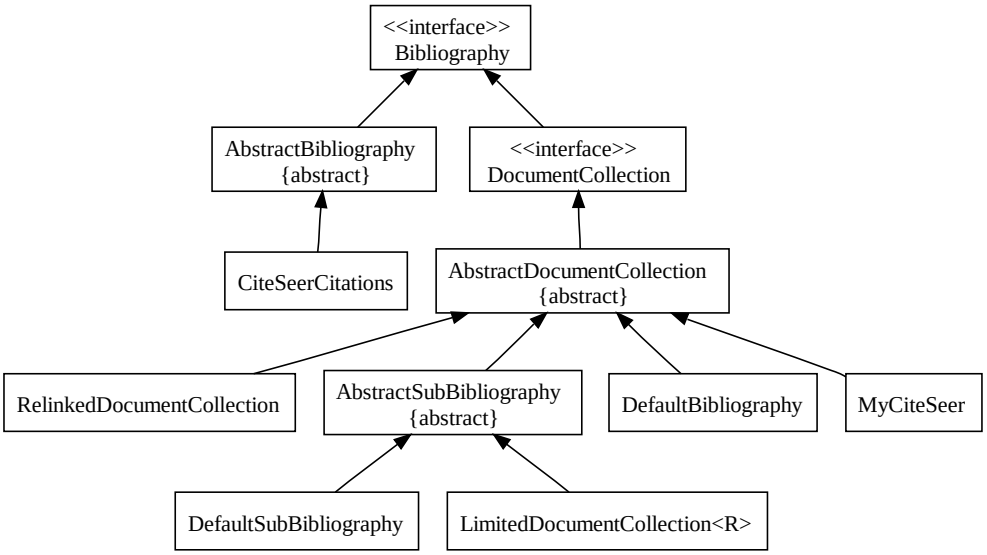


Abbildung 4.8.: Klassendiagramm für die öffentlichen `Bibliography`-Klassen

CiteSeerCitations ist ein Beispiel für die allgemeinste Art von Bibliographien. Sie stellt einen Wrapper zu dem Webfrontend von CiteSeer dar und enthält insbesondere keine Möglichkeit alle Dokumente aufzulisten.

DefaultBibliography ist eine frei veränderbare Dokumentensammlung mit einem `DefaultLinkManager` zur Verwaltung der ebenfalls frei veränderbaren Zitationsrelationen.

MyCiteSeer ist ein Wrapper um auf Dokumentensammlungen in MySQL-Datenbanken (nur lesend) zuzugreifen. Diese Klasse ermöglicht auch sehr große Kollektionen in Citana zu benutzen, welche möglicherweise nicht komplett in den verfügbaren Hauptspeicher passen. Für die Datenbankstruktur siehe Anhang C.

DefaultSubBibliography ist eine Dokumentenkollektion, welche eine Untermenge einer anderen Bibliographie darstellt. Der einzige Unterschied zur `DefaultBibliography` ist, dass diese Klasse keinen eigenen `LinkManager` besitzt, sondern diesen von der Elternbibliographie übernimmt.

RelinkedDocumentCollection ist in gewisser Weise die Umkehrung einer `DefaultSubBibliography`. Hier wird die komplette Dokumentensammlung übernommen aber ein eigener `LinkManager` kann benutzt werden.

LimitedDocumentCollection stellt eine geordnete Liste von Dokumenten mit einem jeweils zugeordneten Rang dar. Zusätzlich kann die Sammlung in der Größe und der Ranggüte begrenzt werden. Diese Klasse wird beispielsweise benutzt um Such- oder Matchingergebnisse zu sammeln und zu verwalten. Die Funktionsweise der Einschränkung entspricht der Beschreibung in Abschnitt 3.1.5 zur Duplikatsuche.

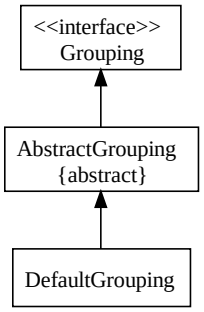


Abbildung 4.9.: Klassendiagramm für die öffentlichen `Grouping`-Klassen

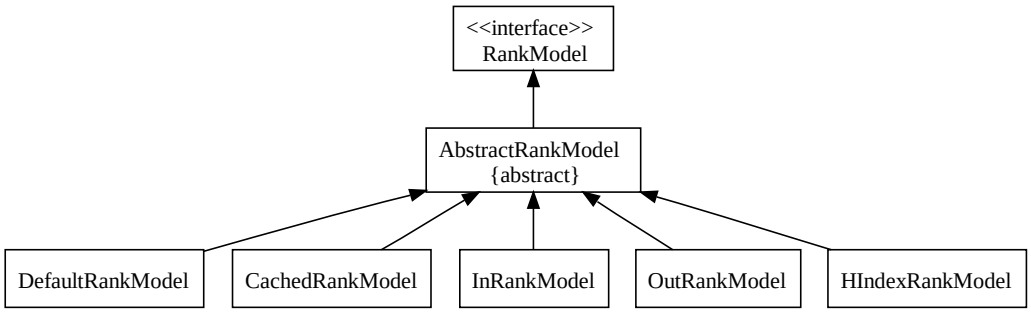


Abbildung 4.10.: Klassendiagramm für die öffentlichen `RankModel`-Klassen

4.2.4. Grouping

`Grouping` verwalten eine Menge von Unterbibliographien einer Dokumentensammlung. An die enthaltenen `DocumentCollection`s werden keine weiteren Forderungen gestellt; im Besonderen dürfen sie sich auch überlappen. Die Implementierung beschränkt sich derzeit auf die Klasse `DefaultGrouping` (vgl. Abbildung 4.9), welche Methoden zum Hinzufügen und Entfernen von beliebigen `DocumentCollection`-Objekten bereitstellt.

4.2.5. RankModel

Ein `RankModel` liefert zu identifizierbaren Objekten (z. B. `Document`-Objekte) einen oder mehrere benannte Rang-Gewichte. Benutzt werden diese um bspw. die berechneten PageRank-Bewertungen oder die Hubs- und Authority-Gewichte für eine Dokumentensammlung zu verwalten. Ein `RankModel` wird aber auch dazu benutzt um Bewertungen von Dokumentensammlungen (z. B. h-Index) zu liefern.

Es sind folgende öffentliche Implementierungen verfügbar (vgl. Abbildung 4.10):

DefaultRankModel ist ein frei änderbares `RankModel`, welches die Rang-Gewichte für beliebige identifizierbare Objekte speichert. Dieses wird hauptsächlich für Bewer-

tungen benutzt, welche aufwendiger sind und die Bewertungen nicht nur für einzelne Objekte seperat berechnen kann.

CachedRankModel bietet für ein anderes `RankModel` eine Caching-Funktionalität zur Erhöhung der Geschwindigkeit, falls dessen Berechnungen umfangreicher sind.

InRankModel stellt eine Rangordnung der Dokumente einer Bibliographie auf Basis der aus der Bibliographie kommenden Zitationen dar. Genauer werden zwei Bewertungen erzeugt: Die Erste ist die Anzahl der aus der Sammlung eingehenden Links und als weitere Bewertung die Summe der Gewichte dieser Links.

OutRankModel stellt eine Rangordnung der Dokumente einer Bibliographie auf Basis der Anzahl der Referenzen in der Bibliographie dar. Analog zum `InRankModel` werden die Anzahl der Links auf in der Sammlung enthaltenen Dokumente sowie deren summierten Gewichte als Bewertungen erzeugt.

HIndexRankModel liefert zu den Dokumentensammlungen einer Gruppierung verschiedene Bewertungen wie sie auch in Abschnitt 3.4 beschrieben werden.

4.3. Aufbau der GUI

Die graphische Benutzeroberfläche ist auf drei Singleton-Klassen mit statischen Zugriffsmethoden aufgebaut, damit diese von beliebigen Objekten benutzt werden können. `CitanaFrame` ist das Hauptfenster der Anwendung und wird von anderen Klassen hauptsächlich benötigt um neu erstellten Dialogen eine korrekte Elternkomponente zu liefern und daran auszurichten. Die Klasse `Output` ermöglicht verschiedenartige Meldungen in einem Textfeld oder in der Konsole auszugeben. Die wichtigste der drei Klassen stellt der `ViewManager` dar, welcher mit seinen zugehörigen Klassen hier vorgestellt wird.

4.3.1. ViewManager

Der `ViewManager` verwaltet die vorhandenen bibliographischen Daten und insbesondere deren Ansichten in einer Baumstruktur. Diese Struktur wird durch Objekte der Klasse `ViewNode` gebildet und in einem `ViewTree` (ein leicht modifizierter `JTree`) angezeigt.

4.3.2. ViewNode

Ein `ViewNode` ist ein Knoten im `ViewTree`, welcher eine Ansicht seiner Daten bereit stellt. Die Ansicht selbst kann eine beliebige `JComponent` sein, welche dann bei einer Selektion des Knoten im `ViewTree` angezeigt wird.

4.3.3. NodeAction

Eine `NodeAction` ist eine Aktion, welche abhängig von einem `ViewNode` ist. Anwendung finden diese Aktionen in den Kontextmenüs der einzelnen `ViewNode`s aber auch in der Menüleiste und der Symbolleiste des Hauptfensters. Für die beiden letzteren Fälle werden bei Änderung des angezeigten Knotens im `ViewTree` für alle Aktionen der neu selektierte Knoten gesetzt. Die Aktionen entscheiden dann anhand des Knotentyps, ob sie ihre Aktion darauf ausführen können oder nicht und sich dementsprechend aktivieren oder deaktivieren.

Die eigentliche Ausführung der zumeist lange dauernden Algorithmen werden in SwingWorker-Klassen implementiert. Die Klasse SwingWorker[1] vereinfacht die Trennung einer langen Operation auf den eigentlichen Worker-Thread und den AWT-Event-Thread, welcher nicht unnötig lang blockiert werden sollte. Die Aktionen benutzen in der Regel einen `CitanaProgressMonitor` zur Anzeige des Fortschritts, Möglichkeit des Abbruchs und auch zur Sperrung aller anderen Funktionen von Citana. Die Sperrung aller Funktionen ist eine einfache aber wirkungsvolle Maßnahme zur Vermeidung von konkurrierenden Objektzugriffen.

4.3.4. AttributedObjectTable

Die auf JTable basierenden Klasse `AttributedObjectTable` ist das wichtigste GUI-Element in Citana um Daten darzustellen. Sie stellt einige Funktionen zur Verfügung, welche es erlauben beliebige, das Interface `Attributed` implementierende Objekte tabellarisch anzuzeigen. Die Schnittstelle `Attributed` enthält Funktionen um mögliche Eigenschaften eines Objekts und deren konkrete Werte abzufragen. In eine `AttributedObjectTable` eingefügt, werden die Objekte schließlich als einzelne Reihen und ihre Eigenschaften in den Spalten dargestellt.

Soweit es möglich ist, sind allgemeine Funktionen für die Tabellenansicht bereits vorimplementiert. Dazu gehört das Sortieren der Inhalte bzgl. beliebiger Spalten, das Ein- und Ausblenden von Spalten, Kontextmenüs mit Exportfunktionen und weitere.

[1] seit Java 6 auch in der API enthalten

5. Experimente mit Citana

5.1. Vergleich der Datenbanken

Die in dieser Arbeit betrachteten Datenbanken sind schon von ihrer Struktur aus sehr unterschiedlich. In Tabelle 5.1 sind einige Kenngrößen zum Vergleich aufgelistet. Abbildung 5.1 zeigt die absoluten Häufigkeitsverteilungen von Einträgen nach Publikationsjahren, soweit diese bekannt sind.

Die Bibliographie LIG besteht ausschließlich aus den Literaturangaben ohne bekannte Beziehungen untereinander oder zu fremden Publikationen. Der Hauptteil der Einträge ist aus den Jahren 1982 bis 2004. Vereinzelt sind auch noch frühere Dokumente verzeichnet.

Der Cora-Datensatz enthält für eine Zitationsdatenbank zwar nur relativ wenige Dokumente, dafür aber sehr viele Referenzen. Die große Anzahl von Zitationen täuscht aber, da in Cora zu den Zitationen bis auf eine Nummer keine weiteren Informationen vorliegen, welche eine Publikation identifizieren könnten (vgl. Anhang C.2.2). Der neuste Dokumenteneintrag ist von 1999, so dass diese Daten für aktuelle Untersuchungen nicht tauglich sind.

Die Datensätze aus CiteSeer, welche als Archiv verfügbar sind, enthalten nur die Daten für die verfügbaren Dokumente und auch nur deren Beziehungen untereinander. Das neuste verzeichnete Dokument ist aus dem Jahr 2006, wobei bereits ein drastischer Rückgang der Einträge für das Jahr 2005 erkennbar ist. Zu der CiteSeer-Datenbank, welche über das Webinterface benutzt werden kann, wird mit Ausnahme der Dokumentenanzahl keine Angabe gemacht.

Die letzte betrachtete Datenbank DBLP ist zwar umfangreich und aktuell, besitzt dafür aber kaum Referenzinformationen.

5.1.1. Skaleninvarianz

In Abbildung 5.2 sind die Verteilungen der Zitierraten resp. die Kanteneingangsgrade im Zitationsgraph für die verschiedenen Zitationsdatenbanken dargestellt. Die Häufigkeitsverteilung $P(k)$ entspricht dieser Darstellung und ist lediglich nach der Gesamtdokumentenanzahl normiert, was bei der Betrachtung hier keine Rolle spielt. In diesem Diagramm sind beide Achsen logarithmisch skaliert, sodass eine Potenzfunktion eine Gerade bildet. Man kann gut erkennen, dass die Meßwerte annähernd einem Potenzgesetz

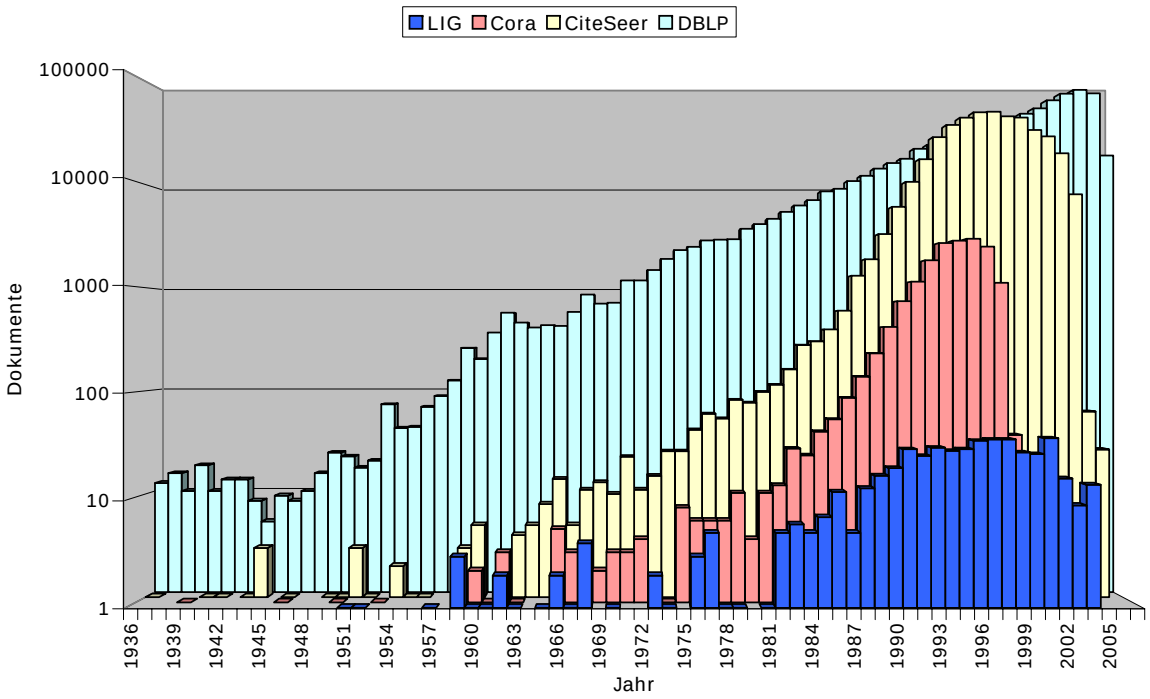


Abbildung 5.1.: Jahresverteilung der Dokumente der verschiedenen Datenbanken für die Jahre 1936 bis 2007 (Stand: 30.6.2007)

Datenbank	**Dokumente**	**Zitationen**	**Beziehungen**
LIG	0	525	0
Cora	36.954	494.351	608.475
CiteSeer	716.772	0	1.751.492
DBLP	0	910.795	112.303

Tabelle 5.1.: Vergleich der Zitationsdatenbanken (Stand: 30.6.2007) Eine Zitation bezeichnet hier ein Dokument, zu welchem kein Volltext vorliegt.

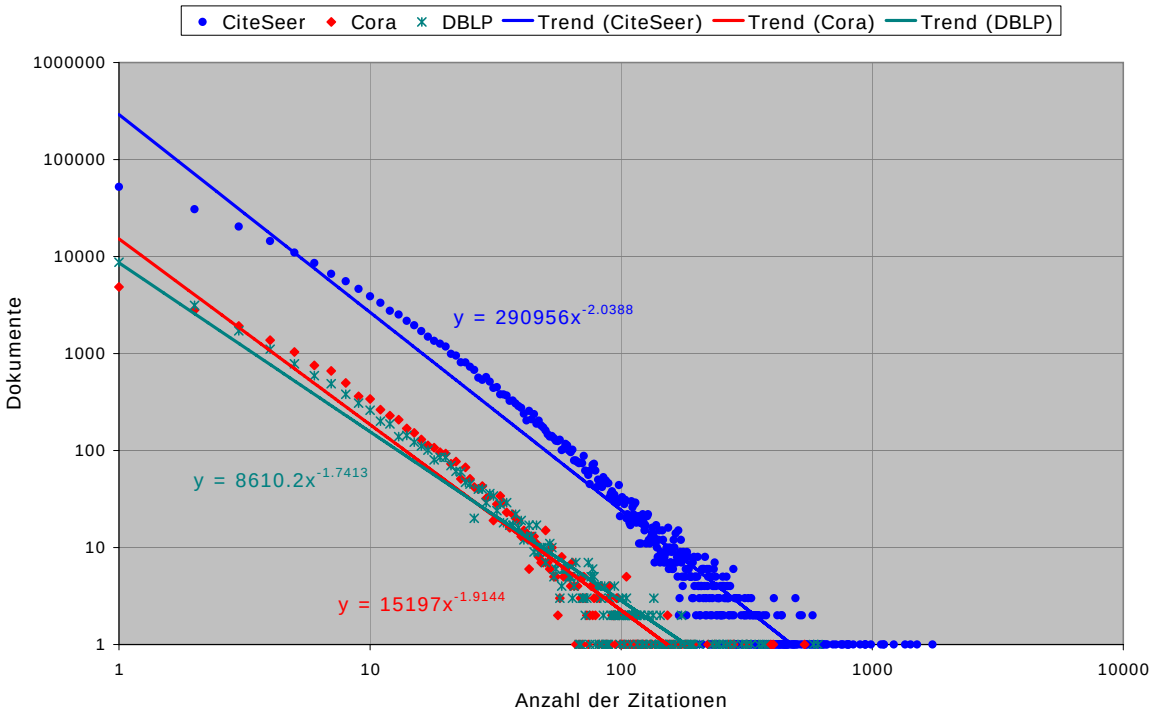


Abbildung 5.2.: Verteilung der Zitierraten (größer Null) in den Zitationsdatenbanken

folgen. Im Weiteren sind dem Trend folgende Potenzfunktionen eingetragen, aus welchen man in etwa $\gamma = 2$ für das Potenzgesetz aus Abschnitt 3.2.1 folgern kann.

Abbildung 5.3 zeigt die entsprechenden Verteilungen der Referenzraten. Hier sieht man direkt, dass diese allgemein keinem Potenzgesetz folgen. Lediglich ab einer gewissen, relativ kleinen Anzahl von Referenzen könnte man für den restlichen Teil des Diagramms eine solche These aufstellen. Die Kurven der Referenzdaten aus Cora und DBLP zeigen deutlich ein innen liegendes Maximum, welches etwa im Bereich von 8 bis 12 Referenzen liegt. Für die Daten aus CiteSeer ist keine derartige Häufung zu erkennen. Das kann damit erklärt werden, dass hier keine vollständigen Referenzinformationen verfügbar sind und das Ergebnis dadurch stark verfälscht wird.

5.2. PHITS in Cora

Zum Testen der verschiedenen Eigenschaften des PHITS-Verfahren wurden einige Versuche mit einem Teil des Cora-Datensatz durchgeführt. Dieser hat den Vorteil, dass die gefundenen Themengebiete mit der vorhandenen Klassifizierung verglichen werden kann. Insgesamt gibt es 70 verschiedene Kategorien in Cora, welche durch zusätzliche Überkategorien eine hierarchische Struktur bilden. Diese wurden größtenteils automatisch anhand der Volltexte ermittelt (siehe [MNRS00]) und können daher auch einige

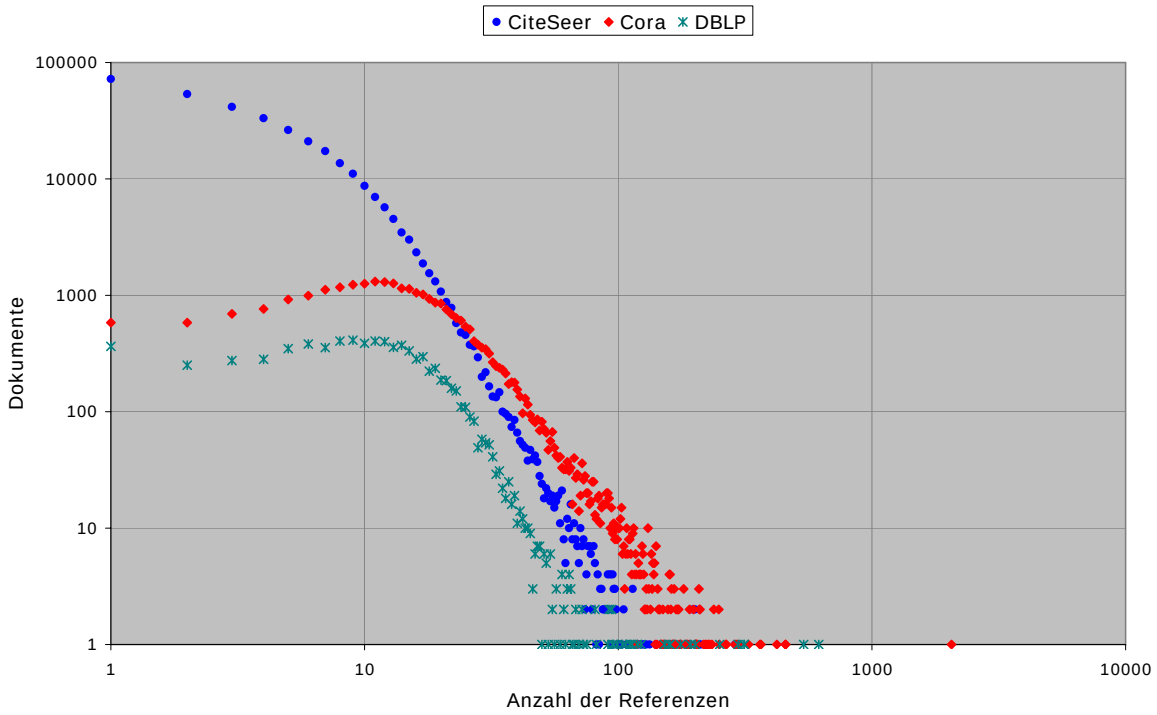


Abbildung 5.3.: Verteilung der Referenzraten (größer Null) in den Zitationsdatenbanken.

Fehler enthalten. Im besonderen kann ein Dokument nicht in einer Überkategorie oder in mehreren Kategorien gleichzeitig eingordnet sein. Diese Kategorien können also nur ein Anhaltspunkt für eine reale Einordnung der Dokumente sein.

5.2.1. Mehrfachzitierungen

Der Cora-Datensatz unterscheidet sich von den anderen Datenbanken u. a. dadurch, dass auch die Anzahl der Zitierungen einer einzelnen Referenz für jedes Dokument verfügbar ist. Für den Zitationsgraph heißt dies, dass eine Multikante auftritt, wenn ein Dokument mehrmals im Text eine Referenz zitiert. In Abbildung 5.4 ist die Häufigkeit für das Auftreten solcher mehrfachen Zitierungen für alle Datensätze und nur für MLx dargestellt. Wie schon bei den skalenfreien Netzen folgt auch diese Verteilung grob einer Exponentialverteilung. Interessant ist, dass mehr siebenfach als sechfach zitierte Referenzen in den Daten vorkommen.[1]

Für die weiteren Untersuchungen werden diese Mehrfachzitierungen nicht benutzt. Die Zitationsgraphen sind reine Linkgraphen, sofern sie nicht anderweitig gewichtet werden.

[1] Eine Vermutung zur Ursache dieses Ergebnisses ist, dass nach siebenmaligem Auftreten die Referenz nicht mehr weiter beachtet wurde und sich damit alle höheren Mehrfachzitierungen anhäufen. Dies müsste aber genauer überprüft werden.

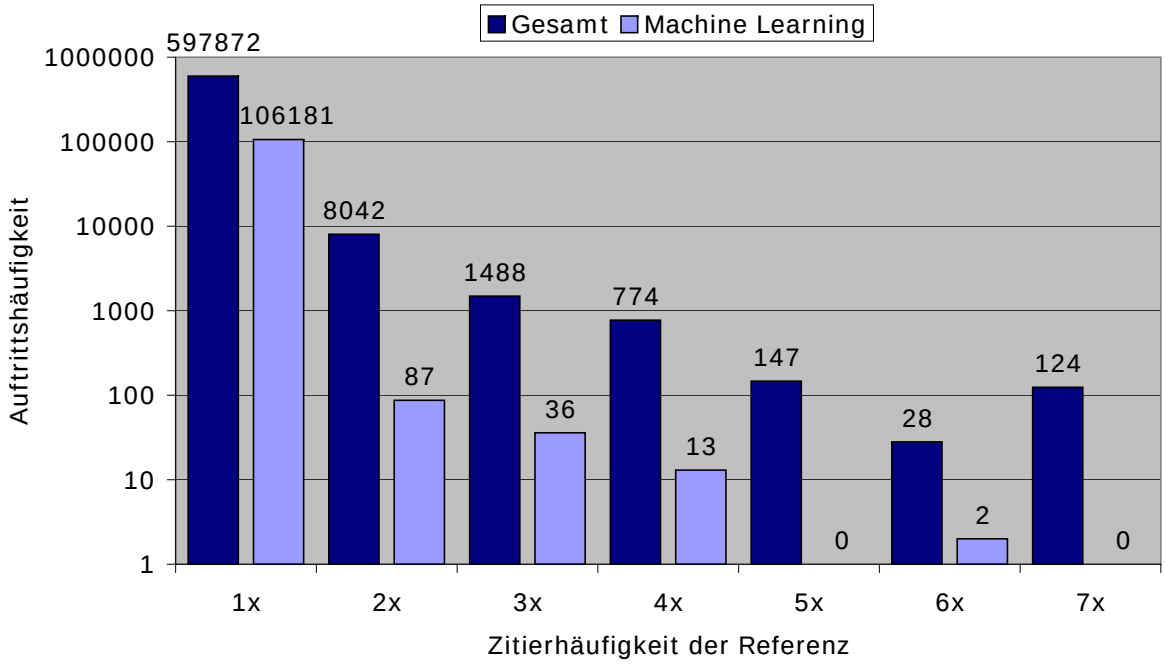


Abbildung 5.4.: Mehrfachzitierungen von Referenzen in Cora

5.2.2. Aufbau eines Zitationsgraphen

Der benutzte Teildatensatz (ML) besteht aus allen Dokumenten der Kategorie */Artificial_Intelligence/Machine_Learning/*, welche in die sieben Unterkategorien *Neural Networks*, *Probabilistic Methods*, *Genetic Algorithms*, *Theory*, *Case-Based*, *Reinforcement Learning* und *Rule Learning* aufgeteilt sind. Insgesamt sind das 4.318 Dokumente, welche mit 33.756 nicht in der Sammlung existierenden Dokumente (davon sind 2.673 Dokument in Cora enthalten und die restlichen 31.083 reine Zitationen ohne Angaben) verbunden sind.

Der von ML induzierte Zitationsgraph ist nicht vollständig zusammenhängend sondern zerfällt in eine Vielzahl von Zusammenhangskomponenten. Wenn jetzt ein Clustering-Algorithmus daraus versucht wenige Kernbereiche zu finden, werden die kleineren Komponenten relativ wahllos einzelnen Bereichen zugeordnet. Daher ist es sinnvoll, mit PHITS nur zusammenhängende Graphen zu betrachten. Um möglichst viele Dokumente aus ML in einem zusammenhängendem Graphen zu erfassen, wird die Menge durch Dummy-Dokumente erweitert. Diese Dummy-Dokumente werden aus den Ergebnissen wieder entfernt, also nicht mitbetrachtet.

Eine solche Erweiterung mit Dummy-Knoten ist in zwei Richtungen sinnvoll: Durch Hinzufügen von Referenzen werden koreferenzierte Dokumente und durch Hinzufügen von Zitationen werden kozitierte Dokumente aus ML verbunden. Die Auswirkung der verschiedenen Erweiterungen auf den Knotenzusammenhang zeigt Tabelle 5.2.

Datensatz	**Anzahl der Knoten**	**Anzahl der Kanten**	**Anzahl der Komponenten**	**Hauptkomponente**
ML	4.318	11.789	753	3.429
ML + Zitationen	6.140	17.317	627	3.575
ML + Referenzen	36.846	92.338	97	4.216
ML + Ref. u. Zit.	38.074	94.530	95	4.218

Tabelle 5.2.: Induzierte Zitationsgraphen und ihre Zusammenhangskomponenten

Zu sehen ist, dass im Besonderen durch Hinzunahme der zitierten Dokumente die Anzahl der Zusammenhangskomponenten im Graph beträchtlich verringert wird. Die größte Zusammenhangskomponente in dem Graph, welcher in beide Richtungen erweitert wurde, enthält 4.218 von insgesamt 4.318 Dokumenten aus ML. Über die verbleibenden 100 Dokumente, welche sich nochmal in 95 Komponenten aufteilen, lässt sich durch graphenbasierte Analysen sicherlich keine bedeutenden Aussagen ableiten, sodass sie aus der zu analysierenden Menge entfernt werden. Im weiteren betrachten wir den Zitationsgraphen MLx, welcher aus der größten Zusammenhangskomponente der durch Zitationen und Referenzen erweiterten Sammlung ML besteht.

5.2.3. Einfluss der Tempering-Parameter

In Abbildung 5.5 sind verschiedenen PHITS-Durchläufe mit dem Ziel ein möglichst gut angepasstes Modell zu erstellen durch die LogLikelihood der Gesamtdaten dargestellt. Die benutzten Parameter dazu waren: Ein Prozent aller Beobachtungen als Tuningsdaten, $\eta = 0{,}9$, β-Verringerung mindestens nach 15 Iterationen sowie 15 finale Iterationen mit $\beta = 1$. Die Tuningsdaten und die Modellinitialisierungen sind für alle Durchgänge genau die gleichen.

Gut sichtbar ist, dass nach der ersten β-Anpassung eine Verflachung der Lernkurve eintritt, welche aber dennoch nach etwa zehn Iterationen die nicht temperierte Kurve übertrifft. Durch das Early-Stopping bekommt man schon ein ziemlich gut angepasstes Modell, wobei es vorteilhaft ist auch die Verschlechterung zuzulassen. Wie man allgemein sieht, sinkt zwar die Likelihood durch einfachere Modelle (also solche mit kleinem β) rapide ab, sie verbessert aber auch die Ausgangsposition für folgende Iterationen mit einem größeren *beta*-Wert. Auf diesen Effekt bauen gerade die Optimierungmethoden, wie das Simulierte Abkühlen, auf.

5.2.4. Stabilität

Was bei einer Vielzahl der durchgeführten Versuche auffällt, ist die relativ große Varianz der erzielten Ergebnisse. Allein durch eine unterschiedliche Initialisierung kann ein Modell entstehen, welches die ursprüngliche Kategorisierung besser oder schlechter erklärt.

In Abbildung 5.7 sind die Klassifizierungen der Authorities einiger, durch unterschiedliche Initialisierungen erzeugten, Modelle dargestellt. Die entsprechenden Lernkurven dieser Modelle zeigt Abbildung 5.6. Die Klassifizierungen zeigen, dass zwar einige relativ stabile Gebiete existieren, aber ebenso andere Gebiete stark abhängig von der Initialisierung sind. Die Lernkurven zeigt ebenfalls, dass unterschiedliche Modellinitialisierungen recht starke Abweichungen zur Folge haben können.

5.2.5. Fazit

Durch die oben gezeigten (und weiteren) Tests können einige Fragen zum Tempered EM aus Abschnitt 3.6.6 beantwortet werden. Die Initialsierung mit einer zufälligen Faktorzuordnung zeigt ein leicht besseres Verhalten als die Initialisierung durch ein vollkommen zufälliges Modell. Die Größe der Tuningsmenge kann relativ klein bleiben (etwa 1 Prozent) um Early-Stopping anwenden zu können. In der Regel ist es besser Modellverschlechterungen einfach zuzulassen. Der Parameter $Ł_{limit}$ braucht nicht weiter betrachtet zu werden und kann einfach bei null belassen, wenn stattdessen spätestens nach einer bestimmten Anzahl von Iterationen β verringert wird. Early-Stopping hat sich ebenfalls als praktikabel erwiesen, sofern β feinstufig genug verringert wird. Abschließend können noch einige finale Iterationen mit dem zuletzt benutzten β-Wert gemacht werden.

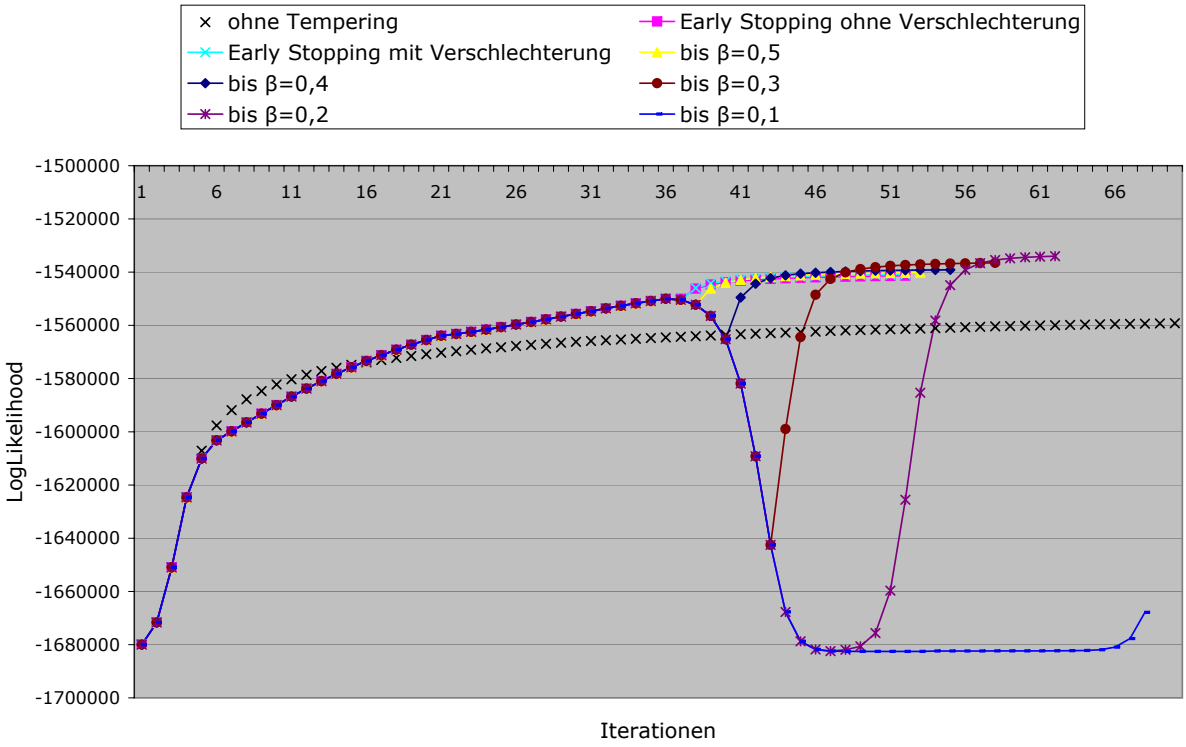


Abbildung 5.5.: Auswirkung von verschiedenen Tempering-Optionen für ein bestangepasstes Modell.

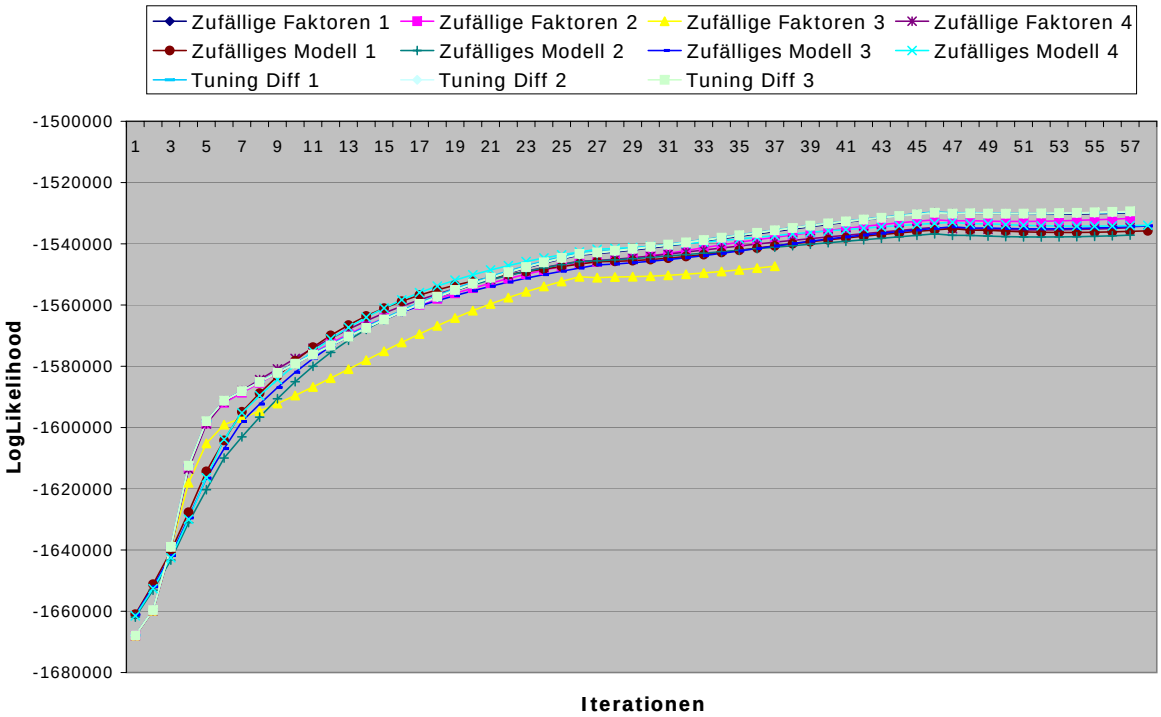


Abbildung 5.6.: Auswirkung von verschiedenen Initialisierungen

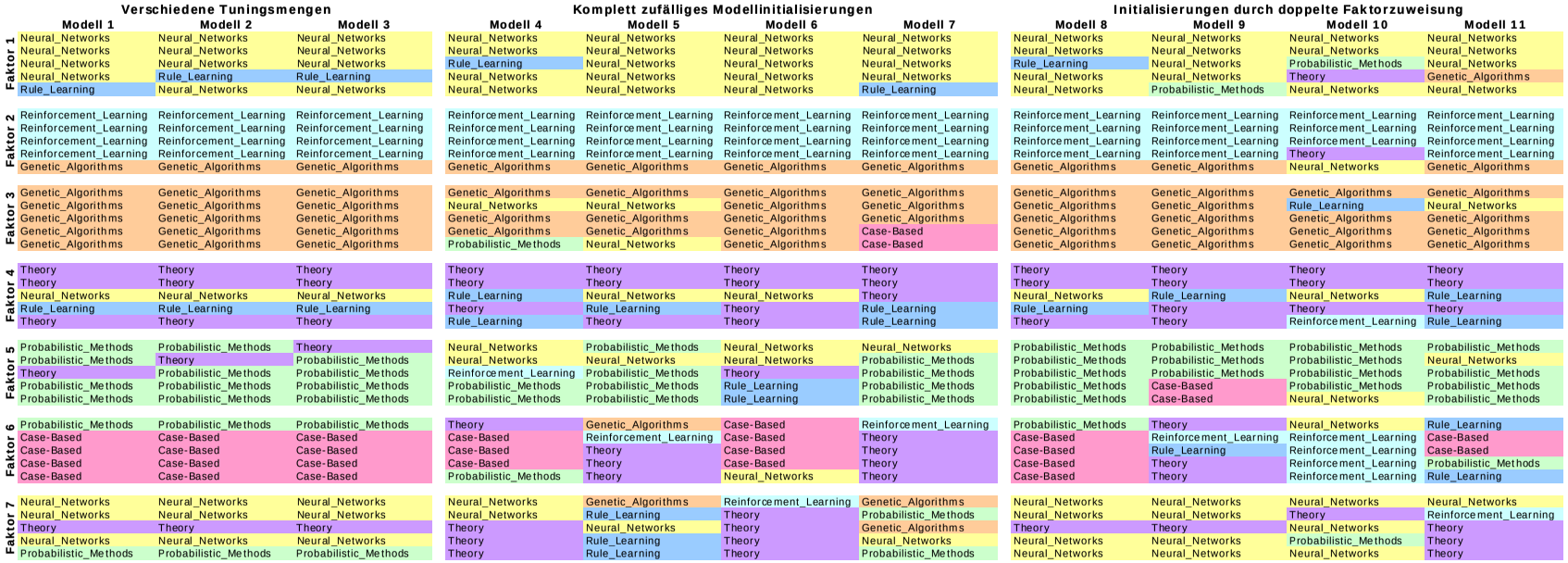


Abbildung 5.7.: Kategorien der Authorithies in MLx. Für jedes berechnete Modell sind bezüglich der einzelnen Faktoren die Kategorien der 5 besten Authories in absteigender Reihenfolge dargestellt. Die Reihenfolgen der Faktoren wurden möglichst passend zueinander neu angordnet.

Spielarten	Reinforcement	Neural Network	Evolutionary	Statistical	EBL	Ensembles	ILP	Inductive	weitere ...	ohne Angabe	Σ
	Lernverfahren										
Go	12	21	14	2	7	0	2	0	2	12	**72**
Chess	18	4	3	2	6	0	8	8	1	6	**56**
Othello	4	15	16	6	1	0	1	1	6	0	**50**
Backgammon	14	14	19	1	0	0	0	0	0	1	**49**
Game Theory	10	0	3	0	0	12	0	0	6	11	**42**
Tic-Tac-Toe	13	6	4	0	1	0	0	1	1	3	**29**
Poker	1	2	5	12	0	0	0	0	0	0	**20**
Checkers	2	2	4	0	0	0	0	0	4	3	**15**
Pig	14	0	0	0	0	0	0	0	0	0	**14**
weitere ...	17	10	8	1	1	0	0	0	6	3	**46**
ohne Angabe	12	4	0	0	0	0	0	0	2	5	**23**
Σ	**117**	**78**	**76**	**24**	**16**	**12**	**11**	**10**	**28**	**44**	

Tabelle 5.3.: Häufigkeitstabelle für die Kategorisierungen nach Spielarten und Lernverfahren. Kategorien mit weniger als 10 Nennungen sind unter „weitere“ zusammengefasst. Mehrfachnennung von Spielarten und Lernverfahren möglich.

5.3. Zitationsanalyse von LIG

5.3.1. Themengebiete

Da für die meisten Einträge in LIG erweiterte Angaben bezüglich des Inhalts verfügbar sind, können direkt Aussagen über die Struktur des abgedeckten Themengebiets gemacht werden. In Tabelle 5.3 sind die beiden thematischen Einteilungen der Dokumente in einer Häufigkeitsmatrix dargestellt.

Deutlich dominieren die Lernverfahren *Bestärkendes Lernen*, *Neuronale Netze* und *Evolutionäre Algorithmen* in der Bibliographie. Die hauptsächlich behandelten Spiele sind *Schach*, *Go* und *Othello*.

Man kann noch versuchen Zusammenhänge zwischen Spielart und Lernverfahren zu finden: Für *Pig* wird auf Basis dieser Daten z. B. nur Verstärkungslernen benutzt; für *Poker* hingegen eher statistische Verfahren. Klare Zusammenhänge kann man aufgrund der relativ geringen Häufigkeiten aber nur schwer ausmachen und stichhaltig begründen.

5.3.2. Eintragssuche in den Zitationsdatenbanken

Der erste Schritt für eine Zitationsanalyse besteht aus der Suche passender Einträge in den Zitationsdatenbanken. Gesucht wurde in den Datensätzen von Cora, CiteSeer und

DBLP durch Dokumentenmatching mit einer Abstandfunktion, welche nur Dokumente mit gleichen oder unbekannten Publikationsjahre vergleicht und eine normalisierte Levenshtein-Distanz zwischen den Titeln berechnet. Die Suche in CiteSeer-Online bestand aus einer normalen Titelsuche, wobei Suchoperatoren wie „and“ und „or“ aus der Anfrage entfernt wurden. Alle gefundenen Einträge wurden anschließend nochmal manuell überprüft. Die Anzahl der gefundenen Einträge in den verschiedenen betrachteten Datenbanken ist in Tabelle 5.4 aufgelistet.

Datenbank	**aus LIG**	**aus DB**	**|D|**	**|C|**	**|D ∪ C|**
Cora	46	43	40	43	43
CiteSeer	176	298	156	104	171
CiteSeer-Citation	241	317	164	317	317
DBLP	158	155	0	10	10

Tabelle 5.4.: Suchergebnisse der insgesamt 525 Einträgen aus LIG in verschiedenen Zitationsdatenbanken. Angegeben werden die Anzahl der gefundenen Einträge aus LIG sowie die Größe der Ergebnismenge. $|D|$ gibt die Anzahl der zitierenden Einträge an. Analog dazu gibt $|C|$ die Anzahl der zitierten Einträge und $|D \cup C|$ die Anzahl der zitierenden oder zitierten Einträge an.

In diesen Suchergebnissen wird die Duplizität der Einträge als ein weiteres Merkmal der verschiedenen Datensätze sichtbar: In der Datenbank CiteSeer wurden mehr Einträge gefunden als es zugehörige Einträge in LIG gibt. Es gibt also zu einigen Dokumente mehrere Einträge in CiteSeer. Umgekehrt wurden in Cora und in der DBLP weniger Einträge als die zugehörige Einträge aus LIG gefunden. Die Mehrfacheinträge in LIG entstehen durch einige mehrfach publizierte aber ansonsten inhaltsgleiche Dokumente mit einem verzeichneten Eintrag je Publikation. Für die CiteSeer-Datenbank können zusätzlich noch Fehler in der automatischen Erfassung hinzukommen.

Es ist sofort ersichtlich, dass die Ergebnisse in Cora und in der DBLP nicht für eine weitere Analyse taugen. In Cora wurden einfach zu wenige Einträge gefunden und im Suchergebnis der DBLP gibt es gerade für 10 der insgesamt 155 gefundenen Einträge Referenzinformationen. Übrig bleiben nur die Suchergebnisse aus CiteSeer: Die 298 Suchergebnisse aus dem Offline-Archiv bezeichnen wir im Folgenden als LigCS und die 317 Sucherergebnisse aus der Online-Suche von CiteSeer nennen wir LigCSC.

Obwohl sich die Größe der Ergebnismenge zwischen LigCS und LigCSC nur gering unterscheidet, wurden 65 LIG-Einträge weniger in der Archiv-Version gefunden. Dies liegt weniger an der fehlenden Aktualität des Archivs, sondern daran, dass die meisten dieser Einträge nur als Zitationen (ohne die eigentlichen Dokumente) in CiteSeer vorhanden sind und diese Informationen nicht im Archiv enthalten sind.

Einen weiteren Unterschied findet man beim Vergleich der Referenz- oder Zitationsinformationen der enthaltenen Einträge. Für LigCSC sind für jeden Eintrag Linkinformationen verfügbar; jedes gefundene Dokument wird von Dokumenten aus CiteSeer-Citation

zitiert und ist somit auch in CiteSeer vernetzt. Für LigCS ist dies nicht der Fall: Nur zu 171 der insgesamt 298 Einträgen sind in irgendeiner Form Linkinformationen verfügbar. Das heißt, dass 127 Einträge aus LigCS für eine Graphenanalyse nicht nutzbar sind, da sie alleinstehende Einzelknoten darstellen.

5.3.3. Aufbau der Zitationsgraphen

Wie schon beim ML-Datensatz sind hier ebenfalls die jeweils aus LigCS und LigCSC gebildeten Graphen nicht zusammenhängend. Etwa die Hälfte der enthaltenen Dokumente bilden eine große Zusammenhangskomponente. Die restlichen Dokumente sind dagegen hauptsächlich isolierte Knoten oder bilden vereinzelt sehr kleine Komponenten mit einer Größe von maximal sieben Knoten.

Die Erweiterung der Ursprungsgraphen hat hier das Hauptziel weitere, noch nicht in der Ursprungsmenge vorhandene, thematisch passende Dokumente aufzufinden. Aus den gleichen Gründen wie in Abschnitt 5.2.2 ist die Bildung eines zusammenhängenden Graphen, ein weiteres Ziel. In Tabelle 5.5 sind einige Eigenschaften der auf unterschiedliche Arten erweiterten Graphen dargestellt.

Datensatz	**Anzahl der Knoten**	**Anzahl der Kanten**	**Anzahl der Komponenten**	**Hauptkomponente**
LigCS	298	266	159	133 (125)
LigCS + Zitationen	1.464	4.081	140	1.323 (148)
LigCS + Referenzen	604	1.341	133	468 (154)
LigCS + Ref. u. Zit.	1.714	6.169	130	1.580 (157)
LigCS + KoRef.	14.044	43.316	1.916	11.935 (150)
LigCS + KoZit.	4.909	16.413	624	4.239 (156)
LigCSC	317	370	161	144 (145)
LigCSC + Zitationen	1.522	5.170	19	1.477 (236)
LigCSC + Referenzen	1.397	4.071	136	1.262 (179)
LigCSC + Ref. u. Zit.	2.559	13.644	9	2.541 (236)

Tabelle 5.5.: Induzierte Zitationsgraphen für LIG-Einträge mit verschiedenen Erweiterungen und ihre Zusammenhangskomponenten in CiteSeer und CiterSeer-Citation. Angaben in Klammern geben die entsprechende Anzahl an Einträgen aus LIG an.

Für die Erweiterungen von LigCS fällt die durchweg hoch bleibende Anzahl von Zusammenhangskomponenten auf. Dies ist hauptsächlich durch die schon bekannten 127 Einzeleinträge in diesem Datensatz begründet, welche eine untere Grenze für die Komponentenanzahl bilden. Wie zu erwarten entstehen bei der Erweiterung mit Koreferenzen oder mit Kozitationen viele neue Zusammenhangskomponenten, welche für eine Zusammenführung nochmal mit Referenzen bzw. Zitationen erweitert werden müssten. Die größte Abdeckung an LIG-Einträgen in den LigCS-Varianten wird durch die Hinzunahme der Referenzen und Zitationen erzielt. Diese Datenmenge wird im Folgenden LigCSx genannt.

Die auf LigCSC basierenden Graphenstrukturen sind im direkten Vergleich denen von LigCS vorzuziehen. Vorteilhaft ist hierfür die größere Abdeckung. Für diese Sammlungsgröße ist es schon nicht mehr praktikabel die Koreferenz- oder Kozitationsgraphen mit Citana zu bearbeiten.[2] Bei den folgenden Betrachtungen wird die größte Zusammenhangskomponente des durch Referenzen und Zitationen erweiterten Zitationsgraphen LigCSC betrachtet, welche wir LigCSCx nennen.

5.3.4. Ranking der Autoren

Die Daten aus LigCSCx sind für eine Betrachtung der Autoren nicht geeignet, da CiteSeer-Citation aus reinen Zitationsangaben nicht die Autorenangaben extrahieren kann. Das hat zur Folge, dass ein großer Teil der Dokumente in einer Autorenanalyse nicht betrachtet werden kann. Hier werden deshalb nur die Daten aus LigCSx benutzt. In Tabelle 5.6 sind die besten 25 Dokumentensammlungen, stellvertretend für die Autoren, bezüglich verschiedener bibliometrischen Indikatoren angeordnet. Die Indikatoren basieren hier auf den allgemeinen Zitierraten in CiteSeer und nicht nur auf Zitationen aus LigCSx.

In den Autorenangaben sind viele Fehler enthalten: Entweder konnten die Autorennamen nicht oder nicht vollständig extrahiert werden oder es werden völlig falsche Autoren ermittelt. Eine kleine Auswahl an „Autoren“ in LigCSx sind z. B. „Sankt Augustin„, „Associate Professor„, „Lehrstuhl Informatik“ oder auch „Un I Vers“ zusammen mit dem Koautor „I Ty Of“. Es kommt auch vor, dass die Prüfer einer Doktorarbeit als deren Mitautoren in CiteSeer eingetragen sind. Die Ergebnisse sind also mit entsprechender Vorsicht zu betrachten.

Bei einem Vergleich mit den meistvertretenden Autoren direkt aus LIG (vgl. Tabelle 5.7) zeigt sich, dass diese in Tabelle 5.6 kaum zu finden sind. Dies lässt sich als eine Fokussierung von LigCSx bezüglich einiger Autoren deuten, welche auch online publizieren.

5.3.5. Monothematisches Dokumentenranking

In diesem Abschnitt sollen die „bedeutenden“ Dokumente für das Gesamtthema „Maschinelles Lernen in Spielen“, welche durch die Zitationsgraphen LigCSx und LigCSCx bestimmt sind, ermittelt werden. Die beiden Datenbanken haben unterschiedliche Eigenschaften, welche sich für einzelne Verfahren besser oder schlechter eignen. Allgemein sind in LigCSCx die Referenzen und in LigCSx die Zitationen vollständiger. Um die Ergebnisse besser untereinander vergleichen zu können, werden in diesem Abschnitt jedoch nur die Ergebnisse für LigCSx vorgestellt. Die entsprechenden Resultate für LigCSCx finden sich in Anhang F.

[2] Dies liegt weniger an der Größe selbst, sondern an der aufwendigen Art wie die Linkinformationen für CiteSeer-Citation in Citana ermittel werden.

	Dokumenteanzahl	n	**Gesamtzitierrate**	c_{tot}	**Relative Zitierrate**	c_{rel}	**h-Index**	h	a
1.	Jordan B. Pollack	37	C. D. Gelatt	1.422	C. D. Gelatt	1.422,0	Robert E. Schapire	13	7,07
2.	Robert E. Schapire	35	M. P. Vecchi	1.422	M. P. Vecchi	1.422,0	Jordan B. Pollack	12	3,91
3.	Xin Yao	23	S. Kirkpatrick	1.422	S. Kirkpatrick	1.422,0	Yoav Freund	10	10,64
4.	Yoav Freund	23	Robert E. Schapire	1.195	Rakesh Agrawal	841,0	Stephen Muggleton	10	8,38
5.	Manfred K. Warmuth	22	Yoav Freund	1.064	Tomasz Imielinski	841,0	Manfred K. Warmuth	9	5,65
6.	Michael L. Littman	20	Rakesh Agrawal	841	Jeffrey Elman	613,0	Jonathan Schaeffer	9	2,49
7.	Risto Miikkulainen	20	Tomasz Imielinski	841	Allen Van Gelder	515,0	Risto Miikkulainen	9	2,38
8.	Yoram Singer	20	Stephen Muggleton	838	John S. Schlipf	515,0	Michael L. Littman	8	10,64
9.	Jonathan Baxter	19	Richard S. Sutton	785	Kenneth A. Ross	515.0	William W. Cohen	8	4,39
10.	Avrim Blum	17	Andrew W. Moore	719	Corinna Cortes	475,0	Yoram Singer	8	4,31
11.	Craig Boutilier	17	Michael L. Littman	681	B. D. Ripley	461,0	Thomas G. Dietterich	7	10,80
12.	Jonathan Schaeffer	17	Jeffrey Elman	613	Glasgow G Ad	345,0	Peter J. Angeline	7	5,69
13.	Manuela Veloso	17	Jordan B. Pollack	563	John Mccarthy	295,0	Craig Boutilier	7	5,31
14.	William W. Cohen	15	Andrew G. Barto	535	Karl Pfleger	281,0	Sebastian B. Thrun	7	5,31
15.	Yishay Mansour	14	Thomas G. Dietterich	529	Cao Feng	264,0	Sebastian Thrun	7	4,08
16.	Alan D. Blair	13	Allen Van Gelder	515	David Zipser	262,0	Xin Yao	7	3,22
17.	Andrew G. Barto	13	John S. Schlipf	515	Scott E. Fahlman	256,0	Jonathan Baxter	7	2,96
18.	Tristan Cazenave	13	Kenneth A. Ross	515	Nick Littlestone	239,0	Tom M. Mitchell	7	2,94
19.	David E. Moriarty	12	Scott E. Fahlman	512	Rodney A. Brooks	224,0	Andrew G. Barto	6	14,86
20.	Duane Szafron	12	Satinder P. Singh	500	Peter Clark	219,0	Satinder P. Singh	6	13,89
21.	Paul J. Darwen	12	John R. Koza	488	Tim Niblett	219,0	David E. Moriarty	6	3,08
22.	Sebastian B. Thrun	12	Corinna Cortes	475	Gerald Tesau	210,0	Richard S. Sutton	5	31.40
23.	Sebastian Thrun	12	B. D. Ripley	461	A. C. Kakas	199,0	Andrew W. Moore	5	28,76
24.	Bruno Bouzy	11	Manfred K. Warmuth	458	F. Toni	199,0	Richard K. Belew	5	8,24
25.	Lex Weaver	11	Rodney A. Brooks	448	R. A. Kowalski	199,0	Jyrki Kivinen	5	6,80

Tabelle 5.6.: Autorenrankings für Dokumente in LigCSx und auf Basis globaler Zitierraten

Autor	Dokumente
Susan L. Epstein	18
David B. Fogel	17
Gerald Tesauro	16
Michael Buro	15
Bruce Krulwich	13
Robert A. Levinson	13
H. Jaap van den Herik	11
Shaul Markovitch	11
Takuya Kojima	11
Jos W. H. M. Uiterwijk	10
Paul E. Utgoff	10

Tabelle 5.7.: Autoren aus LIG mit mehr als 10 Einträgen

Absolute Zitierrate aus ganz CiteSeer		
1422	Optimization by Simulated Annealing. Kirkpatrick et al	1983
841	Mining Association Rules between Sets of Items in Large Databases. Agrawal et al	1993
613	Finding Structure in Time. Elman	1990
548	Learning to Predict by the Methods of Temporal Differences. Sutton	1988
525	Reinforcement Learning I: Introduction. Sutton et al	1998
Absolute Anzahl der Referenzen für ganz CiteSeer		
212	Bibliography of Self-Organizing Map (SOM) Papers: 1998-2001 Addendum. Oja et al	2002
141	Machine Learning and Natural Language Processing. Salgado	2000
133	An Indexed Bibliography of Genetic Algorithms - Papers Available via ftp and www. Alander	1998
129	An Indexed Bibliography of Genetic Programming. Alander	1994
125	An Indexed Bibliography of Genetic Algorithms - Papers Available via ftp and www. Alander	1999

Tabelle 5.8.: Globales Ranking nach Indegree und Outdegree von LigCSx.

Zur Darstellung werden jeweils die fünf höchstbewerteten Einträge tabellarisch für jedes Einzelranking angegeben. Die Einzelrankings werden immer paarweise angegeben: Ein Ranking aufgrund einer Bewertung der Zitationen (Authorities) und eine basierend auf den Referenzen (Hubs). Solche Einträge, welche in den ursprünglichen Suchergebnissen für LIG enthalten sind, sind mit einem „L“ gekennzeichnet. Fehlt diese Markierung, kann das ein Hinweis darauf sein, dass der Eintrag auch in LIG selbst fehlen könnte.

Globales Indegree- und Outdegree-Ranking

Das erste Ranking bewertet die Dokumente aufgrund ihrer Anzahl von Zitationen oder Referenzen unabhängig davon, ob sich diese in einer Themenmenge befinden. Diese Indikatoren sind in der Weboberfläche von CiteSeer die hauptsächlich genutzten Rankingverfahren. In Tabelle 5.8 sind die besten Einträge aus LigSCx für diese beiden Rankings

InWeights			
57,3719	A Decision-Theoretic Generalization of on-Line Learning and an Application to Boosting. Freund et al	1995	L
35,2475	Practical Issues in Temporal Difference Learning. Tesau	1992	L
29,7849	Learning to Predict by the Methods of Temporal Differences. Sutton	1988	
23,8300	Markov Games as a Framework for Multi-Agent Reinforcement Learning. Littman	1994	L
10,2405	Boosting the Margin: A New Explanation for the Effectiveness of Voting Methods. Schapire et al	1997	
OutLinks			
29	Reinforcement Learning: A Survey. Kaelbling et al	1996	
27	Computer Go: an AI Oriented Survey. Bouzy et al	2001	L
25	The Games Computers (and People) Play	2000	L
25	Reinforcement Learning: A Survey. Kaelbling et al	1996	
23	Coevolutionary Search Among Adversaries. Rosin	1997	L

Tabelle 5.9.: Ranking nach Eingangsgewichten für referenzengewichteten Graph sowie nach Referenzenanzahl für LigCSx.

angegeben.

Bei dem Ranking nach Referenzanzahl fällt auf, dass dort hauptsächlich Bibliographien die besten Plätze einnehmen. Diese Einträge sind im Grunde nicht korrekt in CiteSeer erfasst, da die Einträge einer Bibliographie strenggenommen keine Referenzangaben sind.

Themenbezogenes Indegree- und Outdegree-Ranking

Im Unterschied zum vorigen Abschnitt werden beim themenbezogenen Indegree- und Outdegree-Ranking nur Zitationen und Referenzen des eigentlichen Zitationsgraphen gezählt. Hier werden die Summe der Eingangsgewichte des referenzgewichtete Graphen statt einer reinen Zählung der Zitationen angewendet. Jedes Dokument bekommt also nur eine „Stimme“, welche sich auf seine Referenzen gleichmäßig aufteilt (analog zur Übergangsmatrix in PageRank). Die zur Gewichtung nötigen Referenzinformationen sind in LigCSx nicht vollständig, sodass hier LigCSCx genauer betrachtet wird.

In Tabelle 5.9 sind die besten eingangsgewichteten Dokumente sowie die Dokumente mit den meisten thematischen Referenzen für LigCSx dargestellt. Man sieht, dass der überwiegende Anteil der gut gerankten Dokumente bereits in LIG enthalten sind. Dies leuchtet auch ein, da eben diese Dokumente zur Erweiterung von LigCS genutzt wurden und dadurch auch alle verfügbaren Referenzen und Zitationen in LigCSx enthalten sind. Umso interessanter sind dadurch die nicht in LIG enthaltenen Einträge als sehr stark aus dem Themenbereich zitierte Publikationen.

PageRank			
0,0054	Strategy Learning with Multilayer Connectionist Representations. Anderson	1987	
0,0035	Learning to Predict by the Methods of Temporal Differences. Sutton	1988	
0,0035	A Decision-Theoretic Generalization of on-Line Learning and an Application to Boosting. Freund et al	1995	L
0,0033	Practical Issues in Temporal Difference Learning. Tesau	1992	L
0,0023	The Weighted Majority Algorithm. Littlestone et al	1992	
PageRank auf invertierten Graph			
0,0877	Co-evolutionary Search in Asymmetric Spaces	2000	
0,0623	An Optimized Theory Revision Module. Greiner et al	1995	
2E-9	Exploiting Co-Evolution and a Modified Island Model to Climb the Core War Hill. Corno et al		
2E-9	An Investigation, using Co-Evolution, to Evolve an Awari Player. Kendall, Davis	2002	L
2E-9	Computer Go: an AI Oriented Survey. Bouzy et al	2001	L

Tabelle 5.10.: PageRank auf LigCSx (mit $\alpha = 0{,}15$ und $\epsilon = 2^{-30}$)

PageRank

In Tabelle 5.10 sind die Ergebnisse für die PageRank-Bewertung gezeigt. Als Entsprechung für eine Hubs-Bewertung wird der PageRank auf den invertierten Zitationsgraphen angewendet. Hierbei fällt auf, dass zwei sehr starke „Hubs“ gefunden wurden während die weiteren Einträge um einige Größenordnungen schlechter sind.

HITS und gewichtetes HITS

Das HITS-Verfahren wird neben der normalen Variante (Tabelle 5.11) auch mit einem referenzgewichteten Graphen (Tabelle 5.12), analog zu PageRank, durchgeführt. Die beiden Ergebnisse sind dabei sehr unterschiedlich: Die ungewichtete Version hat offensichtlich als Hauptkomponente das Thema „Boosting“ ausgemacht und bewertet. Das gewichtete HITS findet hingegen eher Einträge zum Gebiet „TD-Learning“.

5.3.6. Untersuchung mit PHITS

Zur Untersuchung von LigCSx mit PHITS werden folgenden Einstellungen benutzt: Es sollen 5 versteckte Faktoren gefunden werden. Initialisiert wird das Modell dadurch, dass jedes Dokument (noch unabhängig seiner Rolle) eine einheitliche Faktorzuordnung bekommt bis auf einen zufälligen Faktor, welcher ein doppeltes Gewicht bekommt. Zum Tempering wird Early-Stopping (mit Verschlechterung) mit $\eta = 0{,}95$ mit einer zufälligen Tuningsmenge von einem Prozent der Gesamtbeobachtungen benutzt. Der Tempering-Faktor β wird spätestens nach 20 Iterationen verringert. Zum Schluß werden noch 2 finale Iterationen auf allen Beobachtungen (mit dem zuletzt benutzten β) durchgeführt.

HITS Authorities			
0,2150	A Decision-Theoretic Generalization of on-Line Learning and an Application to Boosting. Freund et al	1995	L
0,0615	The Strength of Weak Learnability. Schapire	1990	
0,0599	Boosting the Margin: A New Explanation for the Effectiveness of Voting Methods. Schapire et al	1997	
0,0516	Boosting a Weak Learning Algorithm By Majority. Freund	1995	
0,0514	Improved Boosting Algorithms Using Confidence-rated Predictions. Schapire	1999	
HITS Hubs			
0,0049	A Short Introduction to Boosting. Freund et al	1999	
0,0048	A Brief Introduction to Boosting. Schapire	1999	
0,0041	Boosting the Margin: A New Explanation for the Effectiveness of Voting Methods. Schapire et al	1998	
0,0038	An Empirical Comparison of Three Boosting Algorithms on Real Data Sets with Artificial Class Noise. McDonald	2003	
0,0037	An Efficient Boosting Algorithm for Combining Preferences. Iyer	1999	

Tabelle 5.11.: Hubs und Authorities in LigCSx

Weighted HITS Authorities			
0,2899	Practical Issues in Temporal Difference Learning. Tesau	1992	L
0,1909	Learning to Predict by the Methods of Temporal Differences. Sutton	1988	
0,0236	Learning to Act using Real-Time Dynamic Programming. Barto et al	1993	
0,0197	Improving Elevator Performance Using Reinforcement Learning. Crites et al	1996	
0,0191	Temporal Difference Learning of Position Evaluation in the Game of Go. Schraudolph et al	1994	L
Weighted HITS Hubs			
0,0164	Evolution of Neural Networks to Play the Game of Dots-and-Boxes. Weaver et al	1996	L
0,0164	Performance Analysis of a New Updating Rule for TD(lambda) Learning in Feedforward Networks for Position Evaluation in Go Game. Chan et al		L
0,0164	Online Learning with Random Representations. Sutton et al	1993	
0,0164	A Trivial but Fast Reinforcement Controller. Moody et al	1994	
0,0136	Improving Temporal Difference Learning for Deterministic Sequential Decision Problems. Ragg et al	1995	L

Tabelle 5.12.: Referenzengewichtete Hubs und Authorities in LigCSx

Als Ergebnis erhält man die Authority-Bewertungen $P(c|z)$ und die Hubs-Bewertungen $P(d|z)$ für jeden der fünf Faktoren. Die jeweils fünf besten Einträge sind in der Tabelle 5.13 aufgelistet. Die einzelnen Faktoren z sind ebenfalls ihrer Bedeutung $P(z)$ nach geordnet: $P(Faktor1) = 0{,}2544$, $P(Faktor2) = 0{,}2208$, $P(Faktor3) = 0{,}2051$, $P(Faktor4) = 0{,}1668$ und $P(Faktor5) = 0{,}1529$. Faktor 1 ist demnach der einflußreichste Faktor in dem Modell.

Die Ergebnisse zeigen, dass wieder „Boosting“ als größtes Gebiet gefunden wird. Die Authorities von Faktor 1 und die aus HITS in Tabelle 5.11 sind sogar beinahe identisch. Die anderen Gebiete lassen sich nicht ganz so eindeutig benennen, zeigen aber auch einige Gemeinsamkeiten. Für den Faktor 2 fällt bspw. auf, dass eine Publikation gleich viermal als Top-Hub vertreten ist. Diese Mehrfacheinträge dürften das Thema des ganzen Gebiets maßgeblich bestimmen und letztlich auch erklären.

Authorities 1	0.2763	A Decision-Theoretic Generalization of on-Line Learning and an Application to Boosting. Schapire, Freund	1995	L
	0.0657	The Strength of Weak Learnability	1990	
	0.0606	Boosting the Margin: A New Explanation for the Effectiveness of Voting Methods. Bartlett, Schapire, Lee, Freund	1997	
	0.0529	Improved Boosting Algorithms Using Confidence-rated Predictions. Schapire	1999	
	0.0510	Boosting a Weak Learning Algorithm By Majority. Freund	1995	
Hubs 1	0.0115	A Short Introduction to Boosting. Schapire, Freund	1999	
	0.0108	Theoretical Views of Boosting and Applications. Schapire	1999	
	0.0102	A Brief Introduction to Boosting, Schapire	1999	
	0.0092	An Efficient Boosting Algorithm for Combining Preferences. Karger, Dharmarajan	1999	
	0.0080	Ensembles of Learning Machines. Valentini	2002	
Authorities 2	0.0512	Practical Issues in Temporal Difference Learning. Tesau	1992	L
	0.0450	Learning to Act using Real-Time Dynamic Programming. Barto, Singh, Bradtke	1993	
	0.0421	Learning to Predict by the Methods of Temporal Differences. Sutton	1988	
	0.0340	The Weighted Majority Algorithm. Warmuth, Littlestone	1992	
	0.0327	Gambling in a rigged casino: The adversarial multi-armed bandit problem. Cesa-Bianchi, Auer, Schapire, Freund	1995	L
Hubs 2	0.0198	Reinforcement Learning: A Survey. Kaelbling, Littman, Moore	1996	
	0.0172	Reinforcement Learning: A Survey. Kaelbling, Littman, Moore	1996	
	0.0149	Reinforcement Learning: A Survey. Kaelbling, Littman, Moore	1996	
	0.0140	Generalized Markov Decision Processes: Dynamic-programming and Reinforcement-learning Algorithms. Littman	1996	
	0.0136	Reinforcement Learning: A Survey. Kaelbling, Littman, Moore	1996	
Authorities 3	0.1073	Practical Issues in Temporal Difference Learning. Tesau	1992	L
	0.0956	Learning to Predict by the Methods of Temporal Differences. Sutton	1988	
	0.0236	Co-Evolution in the Successful Learning of Backgammon Strategy. Blair, Pollack	1998	L
	0.0222	Temporal Difference Learning of Position Evaluation in the Game of Go. Schraudolph, Dayan, Sejnowski	1994	L
	0.0221	Coevolution of a Backgammon Player. Blair, Pollack, Land	1996	L
Hubs 3	0.0174	Coevolutionary Search Among Adversaries. Rosin	1997	L
	0.0099	Methods for Statistical Inference: Extending the Evolutionary Computation Paradigm. Juillé	1999	
	0.0089	Symbiotic Evolution of Neural Networks in Sequential Decision Tasks. Moriarty	1997	
	0.0088	Knowledge Discovery in Chess Databases: A Research Proposal. Fürnkranz		L
	0.0087	Knowledge Discovery in Chess Databases: A Research Proposal. Fürnkranz		L

Authorities 4	0.1685	Markov Games as a Framework for Multi-Agent Reinforcement Learning. Littman	1994	L
	0.0683	Reinforcement Learning I: Introduction	1998	
	0.0661	Multi-Agent Reinforcement Learning: Independent vs. Cooperative Agents. Tan	1993	
	0.0642	Multiagent Reinforcement Learning: Theoretical Framework and an Algorithm. Hu, Wellman	1998	
	0.0472	Reinforcement Learning: A Survey. Kaelbling, Littman, Moore	1996	
Hubs 4	0.0193	Multiagent Reinforcement Learning for Multi-Robot Systems: A Survey. Gu, Yang	2004	
	0.0180	Multiagent Reinforcement Learning: Stochastic Games with Multiple Learning Players. Chalkiadakis	2003	
	0.0174	Reinforcement Learning by Policy Search. Peshkin	2001	
	0.0114	Recent Advances in Hierarchical Reinforcement Learning. Barto, Mahadevan	2003	
	0.0113	Recent Advances in Hierarchical Reinforcement Learning. Barto, Mahadevan	2003	
Authorities 5	0.0474	Coevolving High-Level Representations. Pollack, Angeline	1994	L
	0.0275	Learning Models of Intelligent Agents. Carmel, Markovitch	1996	L
	0.0255	On Evolving Robust Strategies for Iterated Prisoner's Dilemma. Darwen, Yao	1995	L
	0.0228	Inductive Logic Programming. Muggleton	1992	
	0.0222	Efficient Induction Of Logic Programs. Feng, Muggleton	1990	
Hubs 5	0.0249	Computer Go: an AI Oriented Survey. Bouzy, Cazenave	2001	L
	0.0236	The Games Computers (and People) Play. Schaeffer	2000	L
	0.0127	The Challenge of Poker. Davidson, Billings, Szafron	2001	
	0.0126	Learning Logical Exceptions In Chess. Bain	1994	L
	0.0125	Why Co-Evolution beats Temporal Difference learning at Backgammon for a linear architecture, but not a non-linear architecture. Darwen	2001	L

Tabelle 5.13.: PHITS-Ranking mit 5 Faktoren in LigCSx

6. Zusammenfassung und Ausblick

6.1. Zusammenfassung

In dieser Arbeit wurde eine einfaches Verfahren zum Abgleichen von Einträge aus verschiedenen Bibliographien und Zitationsdatenbanken mithilfe der Levenshtein-Distanz vorgestellt. Im Weiteren wurden verschiedene Möglichkeiten zur Erstellung und Analyse von themenbezogenen Zitationsgraphen aufgezeigt.

Daneben wurden die beschriebenen Verfahren in Form des Programms Citana implementiert. Einige Zitationsdatenbanken in unterschiedlichen Formaten wurden in einheitliche MySql-Datenbanken (MyCiteSeer) tranferiert oder direkt über einen Wrapper (CiteSeer-Citation) in Citana eingebunden.

In den Experimenten wurden einige Charakteristiken der unterschiedlichen Datenbanken bestimmt und miteinander verglichen. Um den PHIT-Algorithmus genauer zu untersuchen, wurde es auf vorklassifizierten Daten der Cora-Datenbank angewendet. Anschließend wurden die Einträge aus der Bibliographie LIG in den Zitationsdatenbanken gesucht und aus den Ergebnisse für CiteSeer zusammenhängende Zitationsgraphen erstellt. Abschließend wurden diese analysiert und die Resultate vorgestellt.

Die Ergebnisse der einzelnen Untersuchungen sind durchaus plausibel begründbar. Die relativ schlechten Daten aus den verschiedenen Zitationsdatenbanken lassen eine Beurteilung für LIG selbst aber nicht zu und können nur recht vage Anhaltspunkte liefern.

6.2. Ausblick

In drei verschiedene Bereichen sind Erweiterungen denkbar: In den benutzten Daten, den Analyseverfahren und schließlich in der Umsetzung als Programm.

6.2.1. Zitationsdatenbanken

Grundlegendene Probleme für weitere Analysen sind die Güte und Abdeckung der benutzten Zitationsdatenbanken. Eine Fusionierung der in dieser Arbeit benutzten Datenbanken ist mit entsprechenden Methoden zwar denkbar, aber bezüglich der insgesamt geringen Trefferanzahl von LIG-Einträgen eher wenig erfolgsversprechend. Dazu sind möglicherweise andere Zitationsdatenbanken hilfreicher: Google-Scholar, als Datenbank

mit weitreichender Abdeckung, könnte eine freie Alternative darstellen. Problematisch ist bei ihr, dass zwar die Zitationen zu den einzelnen Einträgen verfügbar sind, aber offensichtlich nicht direkt die Referenzen.

In der Auswertung der Experimente sind teilweise Duplikate eines Dokumentes gemeinsam erschienen. Solche Duplikate können die Auswertungen an sich stören, da so ihre Nachbarschaft im Graphen mehrfach und überbewertet wird. Hier sollte also möglichst noch eine Duplikatenzusammenfassung oder Filterung stattfinden um dies zu verhindern.

Interessant wäre noch eine Untersuchung, inwieweit der komerzielle Science Citation Index im Informatikbereich einen besseren Datenbestand (möglichst ohne die obigen Probleme) als die öffentlichen Datenbanken besitzt.

6.2.2. Analysen

In dieser Arbeit wurde nur ein kleiner Teil von Untersuchungsverfahren für Zitationgraphen behandelt. Im Besonderen wurden nicht alle verfügbaren Daten betrachtet. So wurden zeitliche Informationen oder die Autorenschaften nicht in den graphenbasierten Verfahren mitbetrachtet. Hier könnten bspw. Knoten- oder Kantengewichtungen einbezogen werden, welche abhängig vom Publikationsjahr oder Alter sind. Optimal wäre natürlich noch eine Einbeziehung der Volltexte, welche für den Datensatz LigCSx vollständig verfügbar sind. Als eine Alternative dazu können auch die Zitationskontexte, welche in CiteSeer abrufbar sind (LigCSCx), als textuelle Information zu einer Zitation betrachtet werden. Für solche zusätzlichen Informationen sind dann Multiview-Verfahren interessant. Eine solche Anwendung, welche auf dem Aspekt-Modell aufbaut, ist beispielsweise in [Ord05] beschrieben. Die Einbeziehung von Texten hat zusätzlich den den Vorteil, dass eine Beurteilung der Ergebnisse viel einfacher durch die wichtigen Wörter getroffen werden kann, selbst ohne die Dokumente genauer zu kennen.

Statt dem benutzten PHITS für eine Unterscheidung von Themenbereichen ist eine Hauptkomponentenanalyse (PCA) eine überlegenswerte Alternative. Gerade im Hinblick auf die Schwierigkeiten, welche durch die zufällige Initialisierung und der Anpassung des Tempering-Prozess entsteht, könnte PCA trotz einem falschen Statistikmodell in der Praxis die bessere Wahl sein.

Ein Problem in PHITS ist auch die Festlegung auf eine bestimmte Anzahl zu unterscheidenden Faktoren vor dem Start des Algorithmus. Diese sollten automatisch in den Daten gefunden werden. Hierzu wären zum Beispiel Verfahren analog zu einem hierarchischen Clustering vorzuziehen (siehe [PTL93]).

6.2.3. Citana

Im Laufe der Implementierung von Citana wurden einige Funktionen zurückgestellt um die Komplexität im Rahmen zu halten. Besonders ausgegrenzt wurde hier der Bereich

der Visualisierung. Derzeit gibt es nur eine experimentelle Einbindung des *prefuse visualitation toolkits* zur Anzeige von Zitations- und Gruppengraphen. Hilfreich wären hier ebenfalls die Visualisierung verschiedener Analyseergebnissen. Einige Beispiele finden sich auf der zugehörigen Webseite [10]. Das Ziel müsste sein, jede vorkommende Datentabelle auch graphisch darzustellen.

Im Besonderen könnte noch von einem Parser für verschiedene Operationen profitiert werden. Eine Kantengewichtung könnte damit bspw. frei definiert und getestet werden ohne dazu das Programm selbst ändern zu müssen. Auch eine genauere Filterung oder Gruppierung von Daten, durch frei definierbare Prädikate, kann sinnvoll genutzt werden. Im Weiteren ist eine Loslösung von der doch recht speziellen Zitationsanalyse und eine Verallgemeinerung für Analysen beliebiger Graphen oder Netzwerke sinnvoll, welche aber einige grundlegende Änderungen erforderlich machen.

Dies sind nur einige der wichtigsten Verbesserungsvorschläge. Es bestehen noch sehr viele kleinere Probleme im Programm, welche aber überwiegend Mängel an der Benutzeroberfläche sind.

A. Systemumgebung

A.1. Testsysteme

Entwickelt und getestet wurden die entwickelten Anwendungen auf zwei verschiedenen Systemen:

1. Athlon 64 X2 3800+ mit 1GB RAM unter Windows XP Prof.
2. Mobile AMD Semprom 2800+ mit 512MB RAM unter Windows XP Prof.

A.2. Entwicklungsumgebung

Zur Versionsverwaltung und Backup des Quellcodes nebst zugehörigen Daten wurde Subversion [11] auf einem externen Server verwendet. Neben der Dokumentation von Änderungen am Code ermöglicht der SVN-Server ein gleichzeitiges Arbeiten auf den beiden benutzten Systemen.

Entwickelt wurde Citana mit Suns *Java SE Development Kit* (JDK) in der Version 6 (siehe [12]) und mit Hilfe der Entwicklungsumgebung *Eclipse* (siehe [2]). Als Grafikbibliothek der Benutzeroberfläche wird Javas Swing benutzt. Die Anwendungen sollte somit auf allen Plattformen mit installiertem *Java Runtime Environment* (JRE) ab Version 6 lauffähig sein.

Neben der Standard-API von Java werden folgende zusätzlichen Bibliotheken oder Toolkits benutzt:

Mysql-Connector/J 5.0 ist ein JDBC-Treiber für die Verbindungen zu MySQL-Datenbanken und wird nur von MyCiteSeer-Bibliographien benutzt. [7] Dementsprechend wird diese Bibliothek auch nur benötigt, wenn man mit MyCiteSeer-Datenbanken arbeiten möchte.

Prefuse Ein Toolkit zur Informationsvisualisierung, welches zur Anzeige der Graphenstrukturen in Citana benutzt wird. [10] Diese Funktion ist derzeit nur testweise implementiert.

JGoodies Looks Stellt weitere Look & Feels für Swing zur Verfügung und wird nicht zwingend zur Ausführung der Programme benötigt. [6]

B. Dokumentendistanzen

B.1. Distanzfunktionen in Citana

Equality vergleicht die Dokumententitel auf vollständige zeichenweise Übereinstimmung (unabhängig von der Groß-/Kleinschreibung). Stimmen die Titel zweier Dokumente überein, ist deren Distanz zueinander Null, ansonsten Eins.

Length Difference gibt den (absoluten) Längenunterschied der Dokumententitel als Distanz zurück. Diese Distanz existiert nur zu Testzwecken und ist aufgrund des fehlenden Inhaltsvergleichs keine brauchbare Vergleichsbewertung.

[Uncut|Limit|Limit2] Levenshtein gibt die absolute Edit-Distanz der Dokumententitel zurück. Die Uncut-Version berechnet auch bei einem gesetzten Limit die Edit-Distanz immer vollständig. Die beiden limitierten Varianten implementieren die Abschätzungen aus Kapitel 3.1.5: „Limit“ benutzt Abschätzung (3.5) und „Limit2“ implementiert die Abschätzung (3.6).

Normalized [Uncut|Limit|Limit2] Levenshtein berechnet ebenfalls die Edit-Distanz (in den bereits beschriebenen Varianten) zwischen den Dokumententiteln, wobei diese Distanz noch durch die Titellänge des des Anfragedokumentes normalisiert wird. Diese Funktion ist nicht symmetrisch und damit im strengen Sinn kein Distanzmaß, dafür ist die Berechnung ein wenig einfacher. Da das Anfragedokument für einen Matching-Durchlauf gleich bleibt, können die Distanzen dennoch miteinander verglichen werden.

Normalized [Uncut|Limit|Limit2] Levenshtein With Year gibt eine Distanz von ∞ zurück, falls für beide Dokumente Publikationsjahre angegeben sind und sich diese unterscheiden. Für den Fall gleicher oder fehlender Jahresangaben, wird die normalisierte Levenshtein-Distanz wie in der vorherigen Distanzfunktion ermittelt.

B.2. Laufzeiten

Insgesamt wurden zehn zufällig gewählte Einträge aus LIG in der CiteSeer-Datenbank gesucht. Für neun der zehn Einträge waren die Publikationsjahre bekannt. Die Titel der Sucheinträge hatten die durchschnittliche Länge von 73.5 Zeichen, eine Mindestlänge von 25 Zeichen und eine Maximallänge von 132 Zeichen. Die Limitierung für die Ergebnismengen sind vorgegeben mit $d_{max} = 0{,}5$ und $n_{max} = 5$. Eine Mindestdistanz von 0,5 ist für die normalisierten Levenshtein-Distanzen recht großzügig bemessen, so müssen nur

etwa die Hälfte aller Zeichen in den Titeln übereinstimmen um in die (nicht volle) Ergebnismenge zu gelangen. Für die unnormierte Levenshtein-Distanz erlaubt diese Distanz keinerlei Abweichung und hat damit den gleichen Effekt wie der normale Stringvergleich.

Aus den Laufzeiten in Tabelle B.1 kann man die Wirkung der verschiedenen Limitierungsarten für die (normalisierte) Levenshtein-Distanz abschätzen. Mit den unbeschränkten Levenshtein-Berechnungen haben alle zehn Durchläufen gute 8 Minuten gedauert. Mit der ersten Abschätzung konnten gut 3 Minuten eingespart werden und mit der zweiten Abschätzung lief die Berechnung etwa doppelt so schnell.

Die zweite Methode zur Steigerung der Geschwindigkeit ist der vorherige Vergleich der Jahreszahlen. Hier hat sich für den unbeschränkten Vergleich ebenfalls etwa eine Halbierung der Berechnungszeit ergeben. Die Kombination von Jahresvergleich mit der zweiten Distanzabschätzung hat für die gesuchten Testdaten im Durchschnitt eine Suchlaufzeit von 16 Sekunden, wohingegen die einfache Version 49 Sekunden benötigt. Hochgerechnet auf die 525 Einträge in LIG sind das weniger als 2,5 Stunden gegenüber mehr als 7 Stunden für einen vollständigen Abgleich.

	Durchschnitt	**Min**	**Max**	**Gesamt**
Levenshtein				
Unbeschränkt	00:48.887	00:23.625	01:17.609	08:08.873
Limitierung 1	00:12.771	00:11.703	00:14.047	02:07.719
Limitierung 2	00:12.859	00:11.438	00:14.063	02:08.592
Normalisiertes Levenshtein				
Unbeschränkt	00:49.200	00:24.453	01:18.860	08:12.001
Limitierung 1	00:29.476	00:12.078	00:51.063	04:54.768
Limitierung 2	00:23.895	00:12.016	00:36.032	03:58.955
Normalisiertes Levenshtein mit Jahresvergleich				
Unbeschränkt	00:25.271	00:17.890	00:35.094	04:12.718
Limitierung 1	00:18.890	00:12.297	00:25.109	03:08.908
Limitierung 2	00:16.007	00:12.203	00:19.047	02:40.076
Einfacher Vergleich				
Stringvergleich	00:10.632	00:10.344	00:12.000	01:46.327
Längenunterschied	00:15.674	00:09.641	00:23.062	02:36.748

Tabelle B.1.: Laufzeiten der Distanzfunktionen in Minuten (Athlon X2 3800+)

Es sei noch kurz auf die Laufzeiten der beiden einfachen Vergleichsfunktionen hingewiesen. Der normale Stringvergleich führt am schnellsten zu einer Treffermenge. Die meiste Zeit wird hier auch nicht der eigentliche Stringvergleich benötigen, sondern die Abfrage der kompletten Datenbank. Auf den ersten Blick unerwartet ist der einfache Vergleich der Titellängen um einiges langsamer als ein zeichenweiser Vergleich. Die Erklärung liegt

hier auch nicht bei der Bestimmung des Längenunterschiedes, sondern im Sammeln der Ergebnisse. Dadurch dass Dokumente mit gleicher Titellänge wie das Anfragedokument gefunden werden, wird die Treffermenge sehr schnell sehr groß. Dies liegt daran, dass die Anzahl der Treffer mit der besten Distanz eben nicht durch $n_{max} = 5$ beschränkt werden. Das Einfügen der Treffer ist entsprechend aufwendig, was die relativ hohe Laufzeit erklärt.

C. MyCiteSeer

C.1. DB-Schemata

Das Datenbank-Schema einer MyCiteSeer-Bibliographie besteht aus den DB-Tabellen `document`, `author` und `citation`.

Die Struktur von `document` (Tabelle C.1) erlaubt eine beliebige Anzahl von weiteren Attributen, welche dann als Dokumenteneigenschaften in Citana verfügbar sind. Die Typen dieser Attribute ist zwar grundsätzlich beliebig, es werden aber derzeit nur die jeweiligen Stringrepräsentationen von Citana ausgelesen.

Die `author`-Tabelle (Tabelle C.2) besteht lediglich aus dem Dokumentenbezeichner und einem einzelnen Autorennamen, welche auch beide gemeinsam den Primärschlüssel bilden.

Die Tabelle `citation` (Tabelle C.3) enthält die Bezeichner des zitierenden und des zitierten Dokuments als Primärschlüssel. Es kann zusätzlich auch ein Gewicht, wie beispielweise die Anzahl der Zitierungen, gespeichert werden, falls diese bekannt sind und berücksichtigt werden sollen.

Bei der Erstellung einer eigenen MyCiteSeer-Datenbank sollte darauf geachtet werden, dass für die Felder *citation.cites* und *author.name* ein eigener Index angelegt wird um direkte Zugriffe darauf (beträchtlich) zu beschleunigen. Als Datenbank-Typ sollte MyISAM statt InnoDB gewählt werden, da dieser schneller ist und die Transaktionssicherheit von InnoDB hier nicht benötigt wird (die DB wird von MyCiteSeer nur lesend genutzt).

Feld	**Typ**	**Null**	**Standard**	**Index**
identifier	varchar(255)	Nein		P
title	text	Ja	NULL	
year	year(4)	Ja	NULL	
[Property-Name 1]	[beliebig]	Ja	[beliebig]	
[Property-Name 2]	[beliebig]	Ja	[beliebig]	
⋮	⋮	⋮	⋮	
[Property-Name n]	[beliebig]	Ja	[beliebig]	

Tabelle C.1.: Struktur der Tabelle `document`

Feld	Typ	Null	Standard	Index
document	varchar(255)	Nein		P
name	varchar(255)	Nein		P,I

Tabelle C.2.: Struktur der Tabelle `author`

Feld	Typ	Null	Standard	Index
document	varchar(255)	Nein		P
cites	varchar(255)	Nein		P,I
frequency	double	Nein	1.0	

Tabelle C.3.: Struktur der Tabelle `citation`

C.2. Datenimport

Die als Archive verfügbaren Zitationsdatenbanken stellen unterschiedliche Daten in unterschiedlichen Formaten bereit. Zu den Daten von CiteSeer, CORA und DBLP wurden zu diesem Zweck kleine Hilfsprogramme erstellt, welche ausschließlich zum Import der jeweiligen Datensätze in eine MySQL-Datenbank dienen. Während die einzelnen Tabellen von den einzelnen Programmen selbst erzeugt werden können, ist dies für die Datenbanken nicht möglich. Das heißt, dass die Datenbanken auf eine andere Art (z. B. über phpMyAdmin) mit entsprechenden Zugriffsrechte für den Benutzer erstellt werden müssen.

C.2.1. CiteSeer

Die CiteSeer-Daten liegen in mehreren Dumpfiles vor. Jede dieser Dateien enthält 1000 aneinander gereihte Datensätze, wobei jeder einzelne Datensatz die Metadaten zu einem Dokument im XML-Format beschreibt. Die Daten in den verfügbaren Dateien entsprechen den Antworten, die man über einen Zugriff mittels OAI-PMH (s. a. [8]) von CiteSeer auch erhalten würde. Die Dateien sind in zwei Versionen verfügbar: im Dublin Core Metadaten Standard oder in Dublin Core mit zusätzlichen Metadaten-Einträgen. Beide Archive enthalten offenbar die gleichen Daten und nur die Feldbezeichner unterscheiden sich teilweise. Aufgrund der besseren Bezeichnung von Referenzen wurde die letztere Version zum Import gewählt.

Importiert werden können die Dumpfiles (einzeln) mit dem Programm *CiteSeerImport*. Um alle Dateien gleichzeitig zu importieren, kann das cmd-Skript `allcsimport.cmd` mit der Angabe des Dumpfile-Ordners benutzt werden.

C.2.2. Cora

Der benutzte Datensatz von Cora heißt vollständig „Cora Research Paper Classification“, welcher in [16] verfügbar ist. Die für diese Arbeit interessanten Daten liegen in den Textdateien „classification“, „papers“ und „citations.withauthors“. Inbesonders die sehr umfangreichen Daten im „extraction“-Ordner werden nicht benötigt.

Zum Datenimport dieser Daten in eine MySQL-DB existiert das Programm *CoraImport*. Gestartet werden kann es aus einer Konsole mit

```
coraimport.bat VERZEICHNIS
```

mit der Angabe des Verzeichnis in dem die Cora-Dateien liegen. Weitere optionale Parameter sind:

Kurz	Lang	Standardwert	Beschreibung
`-h`	`--db-host`	`localhost:3306`	Adresse der DB
`-n`	`--db-name`	`cora`	Datenbankname
`-u`	`--db-username`	`root`	Benutzername
`-p`	`--db-password`	- leer -	Passwort

Das Programm erstellt die MyCiteSeer-Tabellen neu, falls diese noch nicht vorhanden sind, leert die Tabellen und füllt sie dann mit den Cora-Daten. Die Referenzdaten in der Tabelle `citation` können Mehrfachkanten enthalten (`frequency` kann größer 1 sein). Wenn das nicht gewünscht ist, reicht die Ausführung des folgenden SQL-Befehls in der Datenbank:

```
UPDATE citation SET frequency = 1;
```

Alternativ können auch in Citana die Linkgewichte fallweise entfernt werden.

C.2.3. DBLP

Die Daten liegen hier in einer einzigen XML-Datei vor. Zum Importieren der DBLP-Daten kann das Programm *DBLPImport* benutzt werden.

C.3. Zitationsanalyse mit SQL

Für sehr große Datenbanken (CiteSeer-Größe) ist die Analyse durch Citana nicht oder nur mit sehr viel verfügbarem Speicher möglich. Grundlegende Statistiken können aber auch direkt durch entsprechende SQL-Abfragen erstellt werden.[1] Als Abfragetool bietet sich *phpMyAdmin* [9] an. Damit können die Ergebnisse u. a. direkt als CSV- oder Excel-Dateien exportiert werden.

[1] Theoretisch könnte man sicherlich auch komplexere Verfahren wie PageRank mit Hilfe von gespeicherten Routinen in SQL implementieren. Fraglich ist aber, ob sich der Aufwand dafür lohnt.

Abgedeckter Zeitbereich

```
SELECT MIN(year), MAX(year)
FROM document;
```

Häufigkeitsverteilung der Publikationsjahre

```
SELECT year, COUNT(identifier) AS n_p
FROM document
WHERE year IS NOT NULL
GROUP BY year
ORDER BY year ASC;
```

Häufigkeitsverteilung der Zitierraten

```
SELECT cited_by_count, COUNT(identifier) AS n_p
FROM (
  SELECT d.identifier,
    count(c.document) AS cited_by_count
  FROM document d LEFT OUTER JOIN citation c
    ON d.identifier = c.cites
  GROUP BY d.identifier
) tmp_table
GROUP BY cited_by_count
ORDER BY cited_by_count ASC;
```

Häufigkeitsverteilung der Referenzraten

```
SELECT reference_count, COUNT(identifier) AS n_p
FROM (
  SELECT d.identifier,
    count(c.cites) AS reference_count
  FROM document d LEFT OUTER JOIN citation c
    ON d.identifier = c.document
  GROUP BY d.identifier
) tmp_table
GROUP BY reference_count
ORDER BY reference_count ASC;
```

Häufigkeitsverteilung der Mehrfachzitierungen

```
SELECT frequency, count(document) AS freq_count
FROM citation
GROUP BY frequency
ORDER BY freq_count ASC;
```

D. CiteSeer-Citation

D.1. Dokumentenbezeichner

In CiteSeer gibt es abhängig von der Eintragsart und teilweise sogar für den gleichen Eintrag unterschiedliche Bezeichner.

Wie schon in der Einleitung erwähnt, muss man in CiteSeer zwischen vorhandenen Dokumenten und den reinen Zitationsangaben unterscheiden. Ein Dokument wird entweder durch eine eindeutige Nummer, welche wir hier #DOC nennen, oder mit einem eher beschreibenden Namen, wie z. B. „buro99from“ bezeichnet. Ein solcher Name kann auch mehrdeutig sein.[1] Zitationsgruppen, welche ja auch Dokumente bezeichnen, haben eine weitere eindeutig bezeichnende Nummer, welche wir #GRP nennen. Das führt nun dazu, dass bis zu drei verschiedene Bezeichner für ein und dasselbe Dokument in CiteSeer existieren.

Zum Arbeiten mit diesen Einträgen muss eine einheitliche und vorallem eindeutige Dokumentenbezeichnung erstellt werden. In der Arbeit mit Citana wollen wir keine grundlegende Unterscheidung zwischen Dokumenten und Zitationsangaben machen und alle Einträge als Dokumente ansehen. Eine einheitliche Bezeichnung in Citana wird daher wie folgt definiert: Für in CiteSeer vorhandene Dokumente ist der neue Bezeichner „csd#DOC“ zu verwenden, also die Dokumentennummer mit dem Präfix „csd“. Ist nur die Zitation verfügbar, soll der neue Bezeichner „csc#GRP“ sein, also die Zitationsgruppennummer mit dem Präfix „csc“.

D.2. Seitenstruktur

Der Wrapper für die Zitationssuche in CiteSeer ist ausgehend von einer Analyse des Webinterfaces von CiteSeer erstellt worden. Als Ergebnis werden drei Anfragen an CiteSeer unterschieden: Zitationssuche, Dokumentenansicht und Zitationskontext. Im Folgenden werden die drei verschiedenen Seitentypen beschrieben. Angaben in {} beschreiben Variablen und solche in [] sind optional.

Zitationssuche

URL: /cs?submit=Citations&q={QRY}&cf={RST}&co={ORD}

[1] Beispiel muß ich noch raussuchen.

Variablen:

QRY	Suchanfrage
RST	Suche einschränken auf „Author“, „Title“ oder „Any“
ORD	Sortierung der Ergebnisse: „Citations“ für Zitierhäufigkeit, „Expected Citations“ für die Zitierhäufigkeit gewichtet nach Jahren oder „Date“

Beschreibung: Liefert die Suchergebnisse für die boolesche Zitationssuche nach QUERY. Die gefundenen Einträge können Dokumente oder auch nicht als Dokument verfügbare Zitationen sein. Standardmäßig werden die Ergebnisse nach der Zitierhäufigkeit sortiert.

Dokumentenseite

URL: /{#DOC|NAME}[.html]

Variablen:

#DOC|NAME	Nummer oder Bezeichner eines Dokumentes

Beschreibung: Liefert ausführliche Informationen zu einem in CiteSeer als Volltext verfügbaren Dokument.

Zitationskontext

URL: /context[summary]/{#GRP}/{#DOC}

Variablen:

#GRP	Nummer der Zitationsgruppe
#DOC	Nummer des Dokumentes. Falls kein Dokument zu der Gruppe vorhanden ist, ist diese 0.

Beschreibung: Zeigt welche Dokumente den spezifierten Eintrag (Dokument oder Zitation) zitieren. Die gelieferten zitierende Dokumente sind zwangsläufig in CiteSeer enthalten und können leicht durch die „CorrectLinks bstimmt werden.

Beispiele:

http://citeseer.ist.psu.edu/context/133/0 listet die zitierende Dokumente einer Zitation mit den einzelnen Zitationskontexten, falls diese vorhanden sind.

http://citeseer.ist.psu.edu/contextsummary/133/0 gibt nur die zitierende Dokumente der Zitation ohne Zitationskontext zurück.

http://citeseer.ist.psu.edu/contextsummary/750/142710 zeigt zitierende Dokumente eines im Volltext vorhandenen Dokuments.

D.3. Implementierung

Die Bibliographie „CiteSeer-Citation“ ist in der Klasse `CiteSeerCitations` implementiert. Bei einer Suchanfrage wird die Zitationssuche mit den entsprechenden Parametern aufgerufen, welche eine Liste von gefundenen Einträge liefert. Die Einträge zeigen auf eine Kontextseite, so dass die Zitationsgruppennummer sowie die Dokumentennummer verfügbar ist. Die Details der Einträge werden für Dokumente aus dem BibTeX-Eintrag der Dokumentenseite oder für reine Zitationen aus der Kontextseite extrahiert. Die Zitationen für einen Eintrag werden über die Kontextseite bestimmt. Die Referenzen werden von der Dokumentenseite ermittelt, wobei hier die Links auf Dokumente auch durch einen alternativen Namen bezeichnet werden. Um die zugehörige Dokumenten- und Gruppennummer zu bestimmen, ist es in diesem Fall immer auch nötig die zugehörige Dokumentenseite abzurufen.

Bei der Zitationssuche und der Abfrage der referenzierenden Dokumente über die Kontextseite werden immer nur die ersten Seiten ausgewertet (in der Regel max. 50 Einträge), welche von CiteSeer schon als „wichtiger“ bewertet werden. Ebenso ist auch die Anzahl der angegebenen Referenzen auf den Dokumentenseiten begrenzt, womit die Komplexität der Abfragen im Rahmen gehalten wird.

Die Seitenabfragen werden derzeit seriell abgearbeitet. Dies ist zwar eine einfache Lösung, aber auch deutlich die langsamste. In der Praxis kommt es häufiger vor, dass einzelne Abfragen sehr lange dauern und dadurch den ganzen Ablauf blockieren. Alle Seitenabfragen werden über die Klasse `WebCache` abgewickelt. Diese stellt als wichtigste Funktion das Cachen von Seiten zur Verfügung. Für jede Anfrage-URL wird die CiteSeer-Antwort als Datei gespeichert. Dabei werden keinerlei evtl. vorhandene Header-Informationen, welche Vorgaben zum Cachen machen, beachtet und der Cache auch nicht automatisch aktualisiert oder gelöscht. Für den Fall, dass die Seite durch einen Fehler nicht abgerufen werden kann (Serverfehler, Timeout), wird die Anfrage bis zu fünfmal wiederholt.

D.4. Methodische Fehler

Bei der Benutzung der Bibliographie CiteSeer-Citation können einige Fehler auftreten, die nicht direkt erklärbar sind. Dies liegt an der Art und Weise wie Citana die Linkinformationen ermittelt und verwaltet. Angenommen Dokument x wird von sehr vielen Dokumenten zitiert, unter anderem auch von y. Werden die Zitationen von x ermittelt, muß y nicht enthalten sein, da wie oben beschrieben nur die wichtigsten Zitationen bestimmt werden. Werden anschließend die Referenzen von y bestimmt, wird die Verbindung „y zitiert x“ bekannt und die Zitationen von x wird durch y erweitert. Die Linkinformationen können sich damit auch während eines Algorithmus ändern und so zu Fehlern führen. Die gleiche Art von Fehlern kann dadurch entstehen, dass nicht alle Referenzen ermittelt werden.

Um solche Fehler zu vermeiden, sollte die Erstellung eines Graphen basierend auf CiteSeer-Citation und die Analyse desselben getrennt vorgenommen werden, indem der Graph als Datei mit den Linkinformationen zwischengespeichert wird.

E. Benutzte Dokumentensammlungen

MyCiteSeer-Datenbanken

CiteSeer aus dem verfügbaren Archiv

Cora Datenbank aus Archiv extrahiert

DBLP Datenbank aus XML-Dateiextrahiert

Referenzinformationen sind, soweit bekannt, in allen MyCiteSeer-Datenbanken enthalten.

als Dateien

lig.bib Die Bibliographie „Maschinelles Lernen in Spielen“. Enthalten sind keine Referenzinformationen.

aus Cora

ML.dc Untermenge aller Dokumente aus der Kategorie „Machinelles Lernen“

MLx.dc ML erweitert durch Referenzen und Zitationen (als Dummy-Dokumente) und davon die größte Zusammenhangskomponente.

aus CiteSeer

LigCS.dc Gefundene LIG-Einträge in CiteSeer. Dokumente haben die Extra-Eigenschaften „lig_Identifier“, „lig_GAME“ und „lig_LEARNING“, welche von den LIG-Einträgen übernommen wurden (dies können auch mehrere sein). Gespeichert sind alle Referenz- und Zitationsinformationen der Dokumente.

LigCSx.dc Die Bibliographie LigCS erweitert um Referenzen und Zitationen und davon die größte Zusammenhangskomponente. Zusätzliche Eigenschaften für die neu hinzugefügten Dokumente sind „x_lig_Identifier“, „x_lig_GAME“ und „x_lig_LEARNING“, welche die entsprechenden Werte durch ihre verlinkenden Dokumente enthalten. Gespeichert sind Referenz- und Zitationsinformationen aller Dokumente.

aus CiteSeer-Citation

LigCSC.dc Gefundene LIG-Einträge in CiteSeer. Dokumente haben die Extra-Eigenschaften „lig_Identifier“, „lig_GAME“ und „lig_LEARNING“, welche von den LIG-Einträgen übernommen wurden (dies können auch mehrere sein). Gespeichert sind Referenz- und Zitationsinformationen aller Dokumente.

LigSCSx.dc Die Bibliographie LigCSC erweitert um Referenzen und Zitationen und davon die größte Zusammenhangskomponente. Zusätzliche Eigenschaften für die neu hinzugefügten Dokumente sind „x_lig_Identifier“, „x_lig_GAME“ und „x_lig_LEARNING“, welche die entsprechenden Werte durch ihre verlinkenden Dokumente enthalten. Gespeichert sind Referenz- und Zitationsinformationen aller Dokumente.

F. Weitere Tabellen

F.1. Rankings für LigCSCx

Absolute Zitierrate aus ganz CiteSeer-Citation			
161	Learning to Predict by the Methods of Temporal Differences. Sutton	1988	
155	Some studies in machine learning using the game of Checkers. Samuel	1967	L
124	Learning from Delayed Rewards. Watkins	1989	
110	Practical Issues in Temporal Difference Learning. Tesauro	1992	L
110	Learning Internal Representations by Error Propagation. Rumelhart et al	1986	
Absolute Anzahl der Referenzen für ganz CiteSeer-Citation			
204	On Growing Better Decision Trees from Data. Murthy		
203	Evolving Artificial Neural Networks. Yao	1999	
203	Wrappers for Performance Enhancement and Oblivious Decision Graphs. Kohavi	1995	
201	Evolving artificial neural networks. Yao	1999	
201	Automated learning of load-balancing strategies for a distributed computer system. Mehra	1992	

Tabelle F.1.: Globales Ranking nach InDegree und OutDegree von LigCSCx.

InWeights			
7,4756	Some studies in machine learning using the game of Checkers. Samuel	1967	L
7,2264	Learning to Predict by the Methods of Temporal Differences. Sutton	1988	
5,7356	Learning from Delayed Rewards. Watkins	1989	
5,5236	Practical Issues in Temporal Difference Learning. Tesauro	1992	L
4,5246	Markov Games as a Framework for Multi-Agent Reinforcement Learning. Littman	1994	L
OutLinks			
144	Computer Go: An AI oriented survey. Bouzy et al	2001	L
142	Learning Logical Exceptions in Chess. Bain	1994	L
98	Coevolutionary search among adversaries. Rosin	1997	L
73	Machine Learning in Computer Chess: The Next Generation. Fürnkranz	1996	L
73	A Game-Learning Machine. Gherrity	1993	L

Tabelle F.2.: Ranking nach Eingangsgewichte für referenzengewichteten Graph sowie nach Referenzenanzahl für LigCSCx.

PageRank			
0,0010	Some studies in machine learning using the game of Checkers. Samuel	1967	L
0,0007	Learning from Delayed Rewards. Watkins	1989	
0,0005	Co-evolving parasites improves simulated evolution as an optimization technique. Hillis	1991	
0,0005	Temporal Difference Learning and TD-Gammon. Tesauro et al	1995	
0,0005	Learning Internal Representations by Error Propagation. Rumelhart et al	1986	
PageRank auf invertierten Graph			
0,0068	Evolutionary Computation and Games. Lucas et al	2006	
0,0044	The Parallel Nash Memory for Asymmetric Games. Oliehoek et al	2006	
0,0043	Reinforcement Learning in Board Games. May	2004	
0,0042	Knowledge discovery in chess databases: A research proposal. Fürnkranz	1997	L
0,0040	Knowledge Discovery in Chess Databases: A Research Proposal. Fürnkranz		

Tabelle F.3.: PageRank auf LigCSCx (mit $\alpha = 0{,}15$ und $\epsilon = 2^{-30}$)

HITS Authorities			
0,0241	Learning to Predict by the Methods of Temporal Differences. Sutton	1988	
0,0210	Some studies in machine learning using the game of Checkers. Samuel	1967	L
0,0176	Practical Issues in Temporal Difference Learning. Tesauro	1992	L
0,0161	Learning from Delayed Rewards. Watkins	1989	
0,0093	Q-learning. Watkins et al	1992	
HITS Hubs			
0,0064	Learning to Solve Markovian Decision Processes. Singh	1993	
0,0064	Reinforcement Learning: A Survey. Kaelbling	1996	
0,0062	Learning Evaluation Functions for Large Acyclic Domains. Boyan	1996	
0,0060	Computer Go: An AI oriented survey. Bouzy	2001	L
0,0058	Automated learning of load-balancing strategies for a distributed computer system. Mehra	1992	

Tabelle F.4.: Hubs und Authorities in LigCSCx

Weighted HITS Authorities			
0,0595	Some studies in machine learning using the game of Checkers. Samuel	1967	L
0,0286	Learning from Delayed Rewards. Watkins	1989	
0,0233	Temporal Difference Learning and TD-Gammon. Tesauro et al	1995	
0,0227	Learning to Predict by the Methods of Temporal Differences. Sutton	1988	
0,0212	Neuro Dynamic Programming. Bertsekas et al	1996	L
Weighted HITS Hubs			
0,0246	Some studies in machine learning using the game of Checkers. Samuel	1967	L
0,0118	Learning from Delayed Rewards. Watkins	1989	
0,0096	Temporal Difference Learning and TD-Gammon. Tesauro et al	1995	
0,0093	NeuroDraughts: the role of representation, search, training regime and architecture in a TD draughts player. Griffith et al		
0,0088	Neuro Dynamic Programming. Bertsekas et al	1996	L

Tabelle F.5.: Referenzengewichtete Hubs und Authorities in LigCSCx

G. Inhalt der DVD

Auf der beiliegenden DVD sind alle elektronisch verfügbaren Daten dieser Diplomarbeit gespeichert. Im Folgenden werden die Inhalte der obersten Verzeichnisse der DVD beschrieben.

Daten enthält die benutzten Ausgangsdaten, bestehend aus der LIG-Bibliographie sowie die benutzten CiteSeer- und CORA-Datensätze im Original und als MySQL-Dumps.

Citana enthält die ausführbaren Dateien von Citana und die Hilfsprogramme.

Sources enthält den Quellcode der erstellten Programme.

Dokumentation enthält diese Ausarbeitung und die generierte Javadoc-Dokumentation des Quellcode.

Webreferenzen

[1] *CiteSeer*. http://citeseer.ist.psu.edu/.

[2] *Eclipse*. http://www.eclipse.org/.

[3] *Google Scholar*. http://scholar.google.de/.

[4] *Graphviz*. http://www.graphviz.org/.

[5] *JabRef reference manager*. http://jabref.sourceforge.net/.

[6] *JGoodies*. http://www.jgoodies.com/.

[7] *MySQL*. http://www.mysql.de/.

[8] *Open Archives Initiative*. http://www.openarchives.org/.

[9] *The phpMyAdmin Project*. http://www.phpmyadmin.net/.

[10] *Prefuse*. http://www.prefuse.org/.

[11] *Subversion*. http://subversion.tigris.org/.

[12] *Suns Java*. http://java.sun.com/.

[13] Feder, Alexander: *BibTeX.org*. http://www.bibtex.org/.

[14] Fürnkranz, Johannes: *Bibliography on Machine Learning in Strategic Game Playing*. http://www.ke.informatik.tu-darmstadt.de/~juffi/lig/.

[15] Ley, Michael: *DBLP Computer Science Bibliography*. http://dblp.uni-trier.de/.

[16] McCallum, Andrew: *Cora Datensätze*. http://www.cs.umass.edu/~mccallum/code-data.html.

Literaturverzeichnis

[AB02] Albert, R. und A. Barabasi: *Statistical mechanics of complex networks*, 2002.

[CC00] Cohn, D. und H. Chang: *Learning to probabilistically identify authoritative documents*. Proceedings of the 17th International Conference on Machine Learning, Seiten 167–174, 2000.

[DLM05] Dong, Peng, Marie Loh und Adrian Mondry: *The „impact factor" revisited*. Biomedical Digital Libraries, 2(1):7, 2005.

[ER90] Egghe, Leo und Ronald Rousseau: *Introduction to Informetrics : quantitative methods in library, documentation and information science*. Elsevier, 1990.

[GBL98] Giles, C. Lee, Kurt Bollacker und Steve Lawrence: *CiteSeer: An Automatic Citation Indexing System*. In: Witten, Ian, Rob Akscyn und Frank M. Shipman III (Herausgeber): *Digital Libraries 98 - The Third ACM Conference on Digital Libraries*, Seiten 89–98, Pittsburgh, PA, June 23–26 1998. ACM Press.

[Hir05] Hirsch, J. E.: *An index to quantify an individual's scientific research output*. PNAS, 102(46):16569–16572, 2005.

[Hof99a] Hofmann, Thomas: *Probabilistic Latent Semantic Analysis*. In: *Proc. of Uncertainty in Artificial Intelligence, UAI'99*, Stockholm, 1999.

[Hof99b] Hofmann, Thomas: *Probabilistic Latent Semantic Indexing*. In: *Proceedings of the 22nd Annual ACM Conference on Research and Development in Information Retrieval*, Seiten 50–57, Berkeley, California, August 1999.

[HP98] Hofmann, Thomas und Jan Puzicha: *Unsupervised Learning from Dyadic Data*. Technischer Bericht TR-98-042, International Computer Science Insitute, Berkeley, CA, 1998.

[Kle99] Kleinberg, Jon M.: *Authoritative sources in a hyperlinked environment*. Journal of the ACM, 46(5):604–632, 1999.

[LBG99] Lawrence, Steve, Kurt Bollacker und C. Lee Giles: *Autonomous Citation Matching*. In: Etzioni, Oren (Herausgeber): *Proceedings of the Third International Conference on Autonomous Agents*, New York, 1999. ACM Press.

[MNRS00] McCallum, Andrew, Kamal Nigam, Jason Rennie und Kristie Seymore: *Automating the Construction of Internet Portals with Machine Learning*. Information Retrieval Journal, 3:127–163, 2000.

[Nav01] Navarro, Gonzalo: *A guided tour to approximate string matching*. ACM Computing Surveys, 33(1):31–88, 2001.

[NBYST01] Navarro, Gonzalo, Ricardo A. Baeza-Yates, Erkki Sutinen und Jorma Tarhio: *Indexing Methods for Approximate String Matching*. IEEE Data Engineering Bulletin, 24(4):19–27, 2001.

[Ord05] Ordyniak, Sebastian: *Probabilistisches Latent Semantisches Indexieren mit mehreren Views für Information Retrieval*. Studienarbeit, Humboldt Universität Berlin, 2005.

[PBMW98] Page, Lawrence, Sergey Brin, Rajeev Motwani und Terry Winograd: *The PageRank Citation Ranking: Bringing Order to the Web*. Technischer Bericht, Stanford Digital Library Technologies Project, 1998.

[PTL93] Pereira, Fernando C. N., Naftali Tishby und Lillian Lee: *Distributional Clustering of English Words*. In: *Meeting of the Association for Computational Linguistics*, Seiten 183–190, 1993.

[RN03] Russell, Stuart und Peter Norvig: *Artificial Intelligence: A Modern Approach*. Prentice-Hall, Englewood Cliffs, NJ, 2nd edition Auflage, 2003.

[Ros98] Rose, K.: *Deterministic annealing for clustering, compression, classification, regression, and related optimization problems*, 1998.

Ehrenwörtliche Erklärung

Hiermit versichere ich, die vorliegende Diplomarbeit ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die aus den Quellen entnommen wurden, sind als solche kenntlich gemacht worden. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Darmstadt, September 2007

Michael Stegbauer