Ähnlichkeitsbasierte Suche in Geschäftsprozessmodelldatenbanken

Andreas Schoknecht

KIT Scientific Publishing

Andreas Schoknecht

Ähnlichkeitsbasierte Suche in Geschäftsprozessmodelldatenbanken

Ähnlichkeitsbasierte Suche in Geschäftsprozessmodelldatenbanken

von
Andreas Schoknecht

KIT Scientific Publishing

Dissertation, Karlsruher Institut für Technologie
KIT-Fakultät für Wirtschaftswissenschaften

Tag der mündlichen Prüfung: 23. März 2018
Referenten: Prof. Dr. Andreas Oberweis, Prof. Dr. Klemens Böhm

Impressum

Karlsruher Institut für Technologie (KIT)
KIT Scientific Publishing
Straße am Forum 2
D-76131 Karlsruhe

www.ksp.kit.edu

Print on Demand 2018 – Gedruckt auf FSC-zertifiziertem Papier

ISBN 978-3-7315-0812-0
DOI 10.5445/KSP/1000084124

Danksagung

Die Ergebnisse dieser Arbeit wurden im Wesentlichen während meiner Tätigkeit als wissenschaftlicher Mitarbeiter am Institut für Angewandte Informatik und Formale Beschreibungsverfahren (AIFB) des Karlsruher Instituts für Technologie (KIT) ausgearbeitet. Für diese Möglichkeit gilt mein besonderer Dank meinem Doktorvater Prof. Dr. Andreas Oberweis. Für die Übernahme des Korreferates bedanke ich mich herzlich bei Prof. Dr. Klemens Böhm sowie bei Prof. Dr. Christof Weinhardt und Prof. Dr. Harald Sack für die Organisation und Durchführung der mündlichen Prüfung.

Bei all meinen Kollegen und Kolleginnen möchte ich mich für die gute Zusammenarbeit bedanken. Insbesondere gilt dieser Dank Meike Ullrich, Stella Möhrle und Uğur Çayoğlu für die interessanten und hilfreichen Diskussionen und Korrekturvorschläge. Zudem möchte ich mich sehr bei Tom Thaler für die Unterstützung mit der RefMod-Miner Plattform sowie den fruchtbaren fachlichen Austausch bedanken. Des Weiteren danke ich allen Studierenden für ihre Arbeit und Unterstützung.

Schließlich bedanke ich mich noch sehr bei meinen Eltern, die mich während meines Studiums und meiner Promotion immer unterstützt und gefördert haben, sowie bei meiner Frau Verena, die mir immer zur Seite stand, insbesondere auch in den kräftezehrenden Zeiträumen mit wenig freier Zeit.

Karlsruhe, im Oktober 2018 *Andreas Schoknecht*

Ähnlichkeitsbasierte Suche in Geschäftsprozessmodelldatenbanken

Zur Erlangung des akademischen Grades eines
Doktors der Ingenieurwissenschaften
(Dr.-Ing.)

bei der Fakultät für Wirtschaftswissenschaften
des Karlsruher Instituts für Technologie (KIT)

genehmigte
Dissertation
von

M.Sc. Inf.-Wirt. Andreas Schoknecht

Tag der mündlichen Prüfung:	23. März 2018
Referent:	Prof. Dr. Andreas Oberweis
Korreferent:	Prof. Dr. Klemens Böhm

Abstract

Die Ausrichtung von Unternehmen wendet sich zunehmend einer prozessorientierten Sichtweise zu. Daher werden Prozesse als wichtiges Mittel angesehen, um die Geschäftsziele zu erreichen. Zur Umsetzung dieser prozessorientierte Sichtweise im Rahmen des Geschäftsprozessmanagements sind Prozessmodelle von großer Relevanz: Sie werden z. B. dazu verwendet Prozesse zu dokumentieren, zu analysieren, zu steuern oder zu verbessern. In Unternehmen ist diese Art der Repräsentation von Abläufen mittlerweile weit verbreitet, wodurch sich teilweise Sammlungen von hunderten oder gar tausenden Geschäftsprozessmodellen ergeben.

Die Erstellung von Geschäftsprozessmodellen erfolgt typischerweise durch ein iteratives Vorgehen, das einen hohen Zeitaufwand sowie hohe Kosten durch eine manuelle, von Grund auf neue Modellierung von Prozessen mit sich bringt. Die Wiederverwendung existierender Modelle bietet sich zur Reduzierung dieser Nachteile an. Allerdings ist das Auffinden von zu einem neu zu modellierenden Geschäftsprozess ähnlichen Modellen durch die großen Modellsammlungen manuell nicht effizient möglich. Hilfreich sind in diesem Fall rechnergestützte Suchmöglichkeiten nach relevanten und ähnlichen Modellen, die als Vorlage zur Modellierung genutzt oder aus denen Teile entnommen werden können. In dieser Arbeit werden verschiedene Ansätze beschrieben, um innerhalb von Prozessmodellbibliotheken zu suchen, sodass ein Modellierer existierende Modelle wiederverwenden kann:

(i) Es werden dazu zwei Varianten vorgeschlagen, wie ähnliche Aktivitäten in Prozessmodellen entdeckt werden können, was als Prozessmodell-Matching bezeichnet wird. Die in dieser Arbeit beschriebenen Triple-S Matching-Verfahren verbessern die Suche nach ähnlichen Aktivitäten in Prozessmodellen. Dabei berücksichtigen die beiden Verfahren die Ähnlichkeit von Aktivitätsbeschriftungen sowie die Graphstruktur der Modelle, um übereinstimmende Aktivitäten zu identifizieren. Eine Evaluation mit zwei weiteren Matching-Ansätzen ergab kein bestes Verfahren. Je nach Datensatz und betrachteter Kennzahl schnitten teilweise die Triple-S Verfahren oder die verglichenen Ansätze besser ab.

(ii) Es wird eine Option zur Suche ähnlicher Prozessmodelle basierend auf einem existierenden Modell vorgestellt. Diese ähnlichkeitsbasierte Suche adaptiert die Latent Semantic Analysis aus dem Information Retrieval für die Suche nach Prozessmodellen. Ein Prozessmodell wird dabei als Textdokument aufgefasst und durch einen Vektor repräsentiert. Die beiden entwickelten LS3-Varianten zur ähnlichkeitsbasierten Suche erzielen in einer Evaluation sehr gute Ergebnisse. Diese Ergebnisse sind teilweise deutlich besser als fünf zum Vergleich herangezogene Suchverfahren.

(iii) Es wird eine grafische Anfragesprache für Prozessmodelle beschrieben, mit deren Hilfe zu einer Anfrage passende Modelle ermittelt werden können. Um nicht nur existierende Prozessmodelle für eine ähnlichkeitsbasierte Suche nutzen zu können, wird bezüglich der dritten Fragestellung eine Anfragesprache definiert, die eine größere Flexibilität bietet. Diese Petri Net Query Model Language ermöglicht eine Beschreibung von (möglicherweise unvollständigen) Prozessmodellen als Suchmuster, zu denen ähnliche Modelle ermittelt werden sollen. Zudem wird ein Algorithmus entworfen, um die Suchergebnisse einer Anfrage unter Berücksichtigung von Prozessmodell-Matching Verfahren zu berechnen.

Inhaltsverzeichnis

Abbildungsverzeichnis

Kapitel 1

Kapitel 2

Kapitel 3

Kapitel 4

Kapitel 5

Kapitel 6

Kapitel 7

Anhang

Tabellenverzeichnis

Kapitel 6

Anhang

Teil I
Hintergrund

Kapitel 1

Einleitung

Die Ausrichtung von Unternehmen wendet sich zunehmend einer prozessorientierten Sichtweise zu (Skrinjar und Trkman 2013). Daher werden Prozesse als wichtiges Mittel angesehen, um die Geschäftsziele von Unternehmen zu erreichen. Durch eine Verbesserung von Prozessen sollen beispielsweise Kosten gesenkt, die Produktivität erhöht und die Zufriedenheit von Kunden gesteigert werden (Kohlbacher 2010; Harmon und Wolf 2014). Um diese prozessorientierte Sichtweise im Rahmen des Geschäftsprozessmanagements umsetzen zu können, sind Prozessmodelle von großer Relevanz. Sie werden etwa dazu verwendet Prozesse zu verstehen, zu analysieren, zu steuern oder zu verbessern (van der Aalst 2013).

Ein *Geschäftsprozessmodell* stellt einen betrieblichen Ablauf vorwiegend in grafischer Form dar und dient beispielsweise zur Dokumentation existierender Abläufe, als Grundlage von Prozessverbesserungsprojekten oder als Basis für eine (teil-)automatisierte Ausführung (Curtis u. a. 1992). Der Nutzen solcher Modelle besteht u. a. aus einem verbesserten Verständnis von Prozessen sowie einer besseren Kommunikation zwischen Prozessbeteiligten über Prozesse, wie in einer Delphi-Studie mit Anwendern, Entwicklern von Modellierungswerkzeugen und Forschern festgestellt wurde (Indulska u. a. 2009). Neben diesen positiven Aspekten wird darüber hinaus die durch Prozessmodelle entstehende Übersicht der Prozesse eines Unternehmens als vorteilhaft bewertet (Kesari u. a. 2003).

In Unternehmen ist diese Art der Repräsentation von Abläufen mittlerweile weit verbreitet (Harmon und Wolf 2014). Dadurch finden sich teilweise Sammlungen von hun-

derten oder gar tausenden Geschäftsprozessmodellen in einem Unternehmen, was in der Literatur auch als ein Teil von „BPM-in-the-Large“ aufgefasst wird (Houy u. a. 2010). Ein kleinerer Modelldatensatz mit 80 Modellen, der die Prozesse von kommunalen Verwaltungen abbildet, ist etwa in (Vogelaar u. a. 2011) beschrieben. Die Suncorp Gruppe, welche Bank- und Versicherungsdienstleistungen anbietet, verwendet sogar mehr als 6.000 Prozessmodelle um ihre Abläufe zu organisieren (Lau u. a. 2011). Zudem enthalten mittlerweile auch Sammlungen von Referenzprozessmodellen einige hundert Modelle. So besteht beispielsweise das SAP Referenzmodell aus mehr als 600 Modellen (Curran und Ladd 1999).

Ein Beispiel eines Geschäftsprozessmodells ist in Abbildung 1.1 dargestellt. Darin ist der Ablauf zur Beantragung einer Dienstreise in der Petri-Netz Notation beschrieben. Um eine Dienstreise zu beantragen, muss zunächst ein entsprechender Antrag von einem Mitarbeiter ausgefüllt und anschließend abgegeben werden. Daraufhin wird der Antrag entweder genehmigt oder abgelehnt und schließlich an den beantragenden Mitarbeiter zurückgegeben.

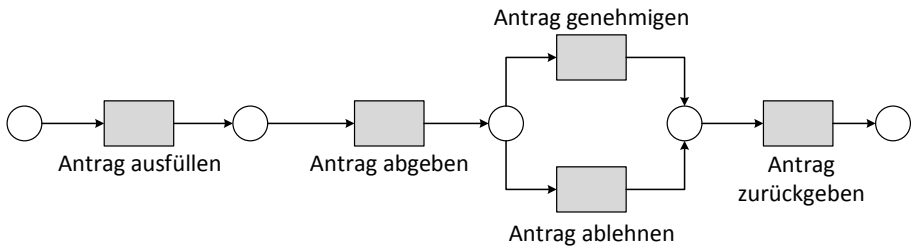


Abb. 1.1: Repräsentation eines Dienstreiseantragsprozesses als Geschäftsprozessmodell in Petri-Netz Notation

Die Erstellung von Geschäftsprozessmodellen erfolgt üblicherweise durch ein iteratives, kommunikations-orientiertes Vorgehen (Hoppenbrouwers u. a. 2005). Zunächst werden etwa durch Workshops, Befragungen oder Beobachtung von beteiligten Mitarbeitern, den sogenannten Fachexperten, die Prozessabläufe ermittelt (siehe etwa (Lübbe 2011; Koch 2011)). Anschließend werden diese Abläufe durch Modellierungsexperten in Geschäftsprozessmodellen dargestellt, die wiederum durch die Fachexperten verifiziert werden. Sollten dabei Unklarheiten oder Fehler in einem Modell gefunden werden, wird das Modell angepasst, bis es letztlich korrekt ist (Wilmont u. a. 2013). Ebenso wird bei Prozessveränderungen, etwa im Rahmen eines Prozessverbes-

serungsprojekts, häufig ein komplett neues Geschäftsprozessmodell erstellt, anstatt beispielsweise auf bereits modellierte Prozesse zurückzugreifen.

Diese Vorgehensweise zur Modellierung von Geschäftsprozessen bringt jedoch einige Nachteile mit sich. So ergeben sich ein hoher Zeitaufwand sowie hohe Kosten durch eine manuelle, von Grund auf neue Modellierung von Prozessen (vgl. etwa (Becker u. a. 2000; Becker u. a. 2010)). Zudem kann diese Vorgehensweise zu Fehlern in den Modellen führen, woraus eine geringe Modellqualität resultieren kann. Mendling u. a. (2007a) stellten etwa fest, dass 10,7 % von 2.003 untersuchten, manuell erstellten Geschäftsprozessmodellen aus der Praxis fehlerbehaftet sind. Potentiell mögliche Optionen zur Reduzierung der genannten Nachteile sind die Wiederverwendung existierender Geschäftsprozessmodelle oder der Einsatz von Process Mining Techniken. Während bei der Wiederverwendung von Geschäftsprozessmodellen Teile oder sogar vollständige Modelle zur Erstellung eines neuen Modells genutzt werden (Koschmider u. a. 2014), versuchen Process Mining Techniken aus Log-Informationen von Informationssystemen automatisiert ein Geschäftsprozessmodell zu generieren, was auch als Process Discovery bezeichnet wird (van der Aalst 2011, S. 10).

In dieser Arbeit wird der Wiederverwendungsaspekt von vollständigen Geschäftsprozessmodellen oder Modellteilen adressiert, d. h., existierende Modelle sollen für die Erstellung eines neuen Geschäftsprozessmodells genutzt werden können. Unter Wiederverwendung wird ein Vorgehen zur Spezifikation, Klassifizierung, Wiederauffindung und Übernahme von Prozessmodellen oder Teilen daraus zur Modellierung verstanden (Koschmider u. a. 2014). Solche existierenden Geschäftsprozessmodelle werden typischerweise in einer *Prozessmodellbibliothek*, auch als Geschäftsprozessmodell-Repository bezeichnet, verwaltet, wobei diese Bibliotheken durchaus mehrere hundert Modelle umfassen können. Somit ist das Auffinden von zu einem neu zu modellierenden Geschäftsprozess *ähnlichen* Modellen manuell nicht effizient möglich. Hilfreich ist in diesem Fall eine Suchmöglichkeit nach relevanten und ähnlichen Modellen, die z. B. als Vorlage zur Modellierung genutzt oder aus denen Teile entnommen werden können.

Ein weiterer Anwendungsfall einer Suchfunktion tritt etwa im Kontext einer Unternehmensfusion auf, sofern die Prozesse vereinheitlicht werden sollen. Eine Suchfunktion ist in diesem Fall hilfreich, da die zu einem gegebenen Modell *ähnlichen Modelle* automatisiert gefunden werden können. Ein zu dem in Abbildung 1.1 dargestellten Beispielprozessmodell ähnliches Modell ist in Abbildung 1.2 wiedergegeben. Diese

beiden Modelle unterscheiden sich lediglich darin, dass im zweiten Modell eine zusätzliche Aktivität ausgeführt wird: Sollte der Dienstreiseantrag genehmigt werden, wird die Personalabteilung darüber informiert. Beispielsweise könnte ein Mitarbeiter Beispielmodell 1 als Sucheingabe verwenden und bekäme als Ergebnis Beispielmodell 2 angezeigt. Daraufhin könnte er sich etwa für eines der Modelle als zukünftig geltenden Ablauf zur Beantragung einer Dienstreise entscheiden oder eines der Modelle ergänzen. In jedem Fall kann er jedoch auf die bereits existierenden Modelle zurückgreifen. Insbesondere muss der Mitarbeiter somit nicht mehr manuell langwierig potentiell hunderte von Modellen selbst auf ihre Ähnlichkeit hin überprüfen, sondern erhält relevante Modelle als Ergebnis einer Suchanfrage.

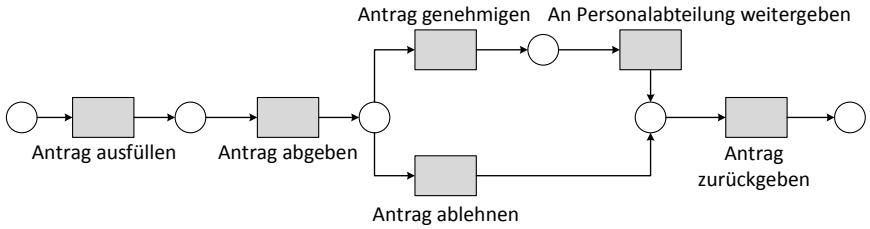


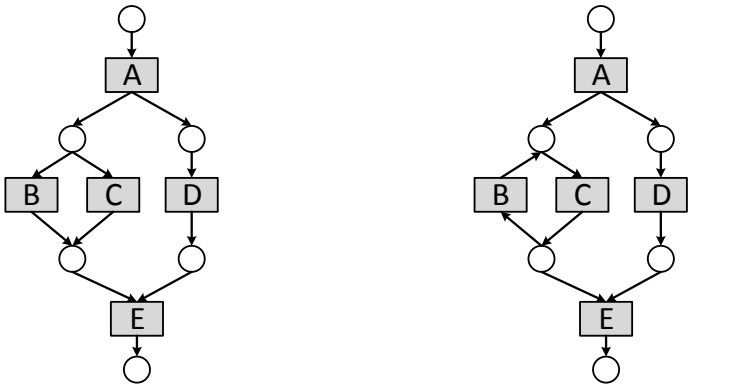
Abb. 1.2: Beispiel eines zu dem Dienstreiseantragsmodell in Abbildung 1.1 ähnlichen Prozessmodells

Die Wiederverwendung von und eine Suchfunktion nach Prozessmodellen werden zudem mit verschiedenen weiteren Vorteilen assoziiert. Zum einen argumentieren Klein und Petti (2006), dass eine Suchfunktion für die Modellierung vorteilhaft sein kann, da einem Modellierer nicht länger nur seine eigenen Gedanken zur Verfügung stehen, sondern er durch andere Modelle neue Erkenntnisse und Ideen erlangen kann. Zum anderen reduziert sich der Zeitaufwand für die Modellierungstätigkeit durch eine Wiederverwendung von Modellen und zusätzlich kann die Qualität von Geschäftsprozessmodellen erhöht werden. Bezüglich des Zeitaufwands beschreiben etwa Hollenbach und Frakes (1996) praktische Erfahrungen bei der Wiederverwendung von Softwareprozessen, die auf eine signifikante Reduzierung der Modellierungszeit hinweisen (im konkreten Fall um 90 %). Weitere Informationen zu den Vorteilen der Wiederverwendung von Prozessmodellen bietet die Übersicht in (Fellmann u. a. 2014). Laut einer Umfrage werden auch in der Praxis über eine Stichwortsuche hinausgehende Suchfunktionalitäten erwartet. So wünschen sich Nutzer von Modellierungswerkzeugen erweiterte Suchmöglichkeiten, da die existierenden Umsetzungen größtenteils hinter den Erwartungen zurückbleiben (Koschmider u. a. 2014).

Die vorgestellten Beispiele illustrieren, wie Suchfunktionen sinnvoll für unterschiedliche Zwecke eingesetzt werden können. Daher werden im Folgenden verschiedene Möglichkeiten beschrieben, um innerhalb von Prozessmodellbibliotheken zu suchen, sodass ein Modellierer zur Unterstützung der Modellierungstätigkeit die in einer solchen Bibliothek vorhandenen Modelle wiederverwenden kann. Erstens werden dazu zwei Varianten vorgeschlagen, wie ähnliche Aktivitäten in Prozessmodellen entdeckt werden können, was als *Prozessmodell-Matching*[1] bezeichnet wird. Zweitens wird eine Option zur Suche ähnlicher Prozessmodelle basierend auf einem existierenden Modell vorgestellt, die sogenannte *ähnlichkeitsbasierte Suche*. Und drittens wird eine grafische *Anfragesprache* für Prozessmodelle beschrieben, mit deren Hilfe zur Anfrage passende Modelle ermittelt werden können. Diese Suche soll allerdings nicht auf einer einfachen Stichwortsuche beruhen, sondern dem *Query-by-Example* Prinzip (Zloof 1977; Zloof 1982) folgen, damit Modellierer auch komplexe Anfragen auf einfache Weise beschreiben können. In diesem Fall bedeutet dies, dass Anfragen ebenfalls Modelle – in einer an Petri-Netze angelehnten Notation – darstellen, sodass Modellierern mit Petri-Netz Kenntnissen die Beschreibung von Anfragen leicht fallen sollte. Zusammenfassend ist in Abbildung 1.4 eine Übersicht der in dieser Arbeit entwickelten Suchmöglichkeiten dargestellt. Eine detailliertere Beschreibung der wissenschaftlichen Fragestellungen erfolgt im folgenden Kapitel 1.1.

Ein wichtiger Aspekt beim Entwurf der Suchfunktionen ist die Berücksichtigung der Ähnlichkeit von Prozessmodellen, um nicht nur Suchergebnisse berechnen zu können, die exakt einer Sucheingabe entsprechen. Dies ist vergleichbar mit einem Teilbereich des *Information Retrieval*, bei dem zu textuellen Sucheingaben nicht nur die Dokumente im Ergebnis vorkommen sollen, die exakt die gesuchten Begriff enthalten, sondern auch Dokumente in denen verwandte Begriffe vorkommen (Manning u. a. 2008, Kapitel 1.1). Übertragen auf die Suche nach Prozessmodellen bedeutet dies etwa, dass Synonyme in den textuellen Beschriftungen berücksichtigt werden müssen. Im zuvor beschriebenen Beispiel der Dienstreiseantragsprozesse ist dies der Fall, wenn in einem Modell anstatt der Beschriftung „Antrag genehmigen“ die Beschriftung „Antrag gestatten“ verwendet wird. Zudem kann sich auch die Struktur bzw. das Ablaufverhalten von Prozessmodellen unterscheiden, selbst wenn die Beschriftungen gleich sind.

[1] In gleichem Sinne wie Matching werden in der Literatur auch die englischen Begriffe *Alignment*, *Mapping* oder *Correspondence* verwendet. In dieser Arbeit wird dafür einheitlich der Begriff Matching genutzt.



(a) Alternative zwischen den Aktivitäten B und C

(b) Beliebig häufige Wiederholung der Aktivitäten B und C

Abb. 1.3: Zwei strukturell sehr ähnliche Modelle, die sich in ihrem Ablaufverhalten deutlich unterscheiden (Modelle in Anlehnung an (van der Aalst u. a. 2006))

Abbildung 1.3 zeigt zwei Modelle, die aus struktureller Sicht nahezu identisch sind. Sie unterscheiden sich lediglich in der Kantenrichtung der Aktivität B, wodurch sich jedoch ein stark geändertes Ablaufverhalten ergibt: In Modell 1.3a werden die Aktivitäten B und C alternativ ausgeführt, während sie in Modell 1.3b beliebig oft hintereinander ausgeführt werden können. Diese Aspekte müssen bei den Suchfunktionen berücksichtigt werden, um einem Modellierer alle für ihn relevanten Modelle zur Wiederverwendung bereitstellen zu können. Weitere Informationen zur Berechnung eines Ähnlichkeitswerts für Prozessmodelle und zwischen Modellelementen werden in Kapitel 2.2 erläutert.

1.1 Betrachtete Forschungsfragen

Wie zuvor beschrieben, ist das Ziel dieser Arbeit der Entwurf unterschiedlicher Suchmöglichkeiten für Geschäftsprozessmodelle in Prozessmodellbibliotheken, um eine Wiederverwendung dieser Modelle zu ermöglichen (siehe Abbildung 1.4). Für diese drei abgebildeten Optionen lassen sich entsprechende Forschungsfragen aufstellen, die mehrere Beiträge für Wissenschaft und Praxis liefern. Diese werden im Folgenden detaillierter ausgeführt:

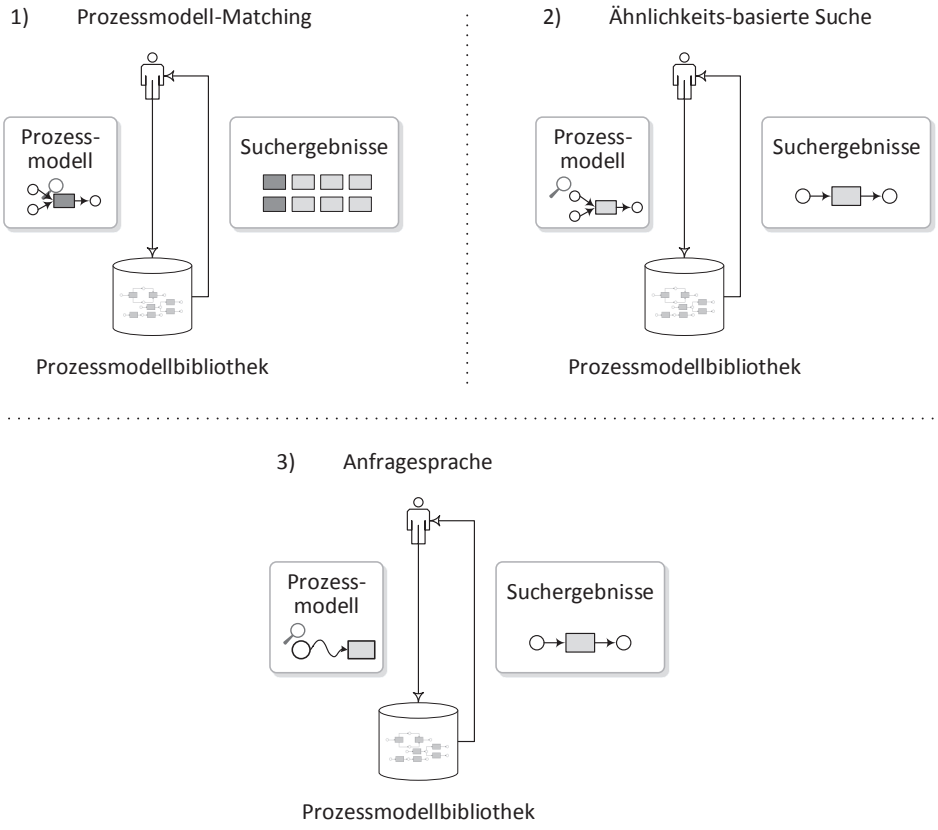


Abb. 1.4: Übersicht der entwickelten Suchmöglichkeiten

(i) Wie können ähnliche Aktivitäten in Prozessmodellen ermittelt werden?
Die Berechnung von Übereinstimmungen zwischen Aktivitäten mehrerer Prozessmodelle (Prozessmodell-Matching) wird für vielfältige Analysezwecke verwendet, da sie häufig als Grundlage eines Ähnlichkeitsmaßes für Prozessmodelle dienen (Schoknecht u. a. 2017b; Thaler u. a. 2017). Solche Techniken des Prozessmodell-Matchings finden sich daher unter anderem zur Erkennung von Unterschieden zwischen Implementierungen eines Geschäftsprozessmodells und eines zugehörigen Referenzmodells (Küster u. a. 2006) oder zur Suche nach Prozessmodellen (Dumas u. a. 2009).

In dieser Arbeit werden zwei Verfahren zur Bestimmung eines Prozessmodell-Matchings in Kapitel 4 beschrieben. Diese können zum einen unabhängig von weite-

ren Modellierungs- oder Analysemethoden eingesetzt werden, um z. B. eine manuelle Analyse von Aktivitäten durch Unternehmensmitarbeiter zu unterstützen. Zum anderen werden sie in Kombination mit der in Kapitel 6.3 beschriebenen Anfragesprache verwendet, um bei der Ergebnisberechnung neben Modellen mit exakt übereinstimmenden Beschriftungen auch relevante Modelle mit ähnlichen Beschriftungen identifizieren zu können. Diese Forschungsfrage bezieht sich damit auf den linken Teil in Abbildung 1.4. Als Eingabe der Suchfunktion wird ein Prozessmodell übergeben, zu dessen Aktivitäten ähnliche Aktivitäten in den Modellen einer Prozessmodellbibliothek gesucht werden.

Die Bestimmung von ähnlichen Aktivitäten in Geschäftsprozessmodellen durch die Berechnung eines Ähnlichkeitsgrads ist bislang noch nicht abschließend gelöst. Bei einem Vergleich verschiedener Ansätze während des Process Model Matching Contests (Cayoglu u. a. 2014) im Rahmen der Business Process Management Konferenz 2013 wurde etwa festgestellt, dass die eingereichten Beiträge qualitativ nicht überzeugen konnten. Die Qualitätsmaße Precision und Recall der einzelnen eingereichten Lösungen lagen jeweils unter 70 %, wobei sich der F-Wert[2] jeweils deutlich unter diesem Niveau befand (höchster Wert 45 %) (Cayoglu u. a. 2014). In dem verwandten Bereich des Datenbankschema-Matchings werden hingegen höhere Werte erzielt (Peukert u. a. 2010). Auch die Untersuchung von Klinkmüller u. a. (2014) weist darauf hin, dass weitere Verbesserungen möglich sind. Dabei wurde ermittelt, inwiefern die Ergebnisse von Ähnlichkeitsberechnungen durch Nutzereingaben verbessert werden können. Die Resultate zeigen schließlich, dass sich durch zusätzliche Informationen von Nutzern eine deutliche Qualitätsverbesserung erzielen lässt.

Beim zweiten Process Model Matching Contest 2015[3] wurden zwar bessere Ergebnisse erzielt, die Qualität der Resultate ist jedoch mehrheitlich nach wie vor unbefriedigend (Antunes u. a. 2015). Bezüglich der Ergebnisse erreichten einige Matching-Ansätze bei diesem Wettbewerb F-Werte bis zu 67 % für einzelne Datensätze, allerdings konnten nur zwei der teilnehmenden Ansätze über alle drei Datensätze hinweg gut abschneiden. Zudem wurden in einem Datensatz 36 % der gesuchten Übereinstimmungen von keinem Matching-Ansatz entdeckt. Ein weiterer Aspekt betrifft die verschiedenartige Herangehensweise der Ansätze. Manche Ansätze vergleichen etwa nur zwei Modelle

[2] Weitere Informationen zu Precision und Recall sowie zum F-Wert finden sich in (van Rijsbergen 1979) bzw. (Euzenat und Shvaiko 2013) oder Kapitel 3.4.
[3] https://ai.wu.ac.at/emisa2015/contest.php

miteinander, während andere einen kompletten Modell-Datensatz untersuchen. Wiederum andere nutzen überwachte Maschinenlernverfahren zur Berechnung der Ergebnisse. Diese sicherlich sinnvollen Ansätze sind jedoch hinsichtlich der Ergebnisse schwierig zu vergleichen, da sie beispielsweise andere Voraussetzungen haben und somit potentiell nicht für alle Anwendungsfälle einsetzbar sind. Dies wäre etwa der Fall, wenn für ein überwachtes Lernverfahren keine Trainingsdaten zur Verfügung stehen oder diese aufwändig zu ermitteln sind.

Aus wissenschaftlicher Sicht stellen die in Kapitel 4 vorgestellten Prozessmodell-Matching Verfahren neuartige Ansätze dar, von denen einer bereits in den Matching Wettbewerben eingesetzt wurde. Aus praktischer Sicht wird durch diese Verfahren die Analyse von Prozessmodellen auf ähnliche Aktivitäten sowie deren Wiederverwendung oder Vereinheitlichung ermöglicht.

(ii) Wie können zu einem Prozessmodell ähnliche Modelle ermittelt werden?
Ähnlichkeitsberechnungen zwischen Geschäftsprozessmodellen dienen oftmals als Basis für Techniken im Bereich der Modellierung von Geschäftsprozessen und der Verwaltung von Modellbeständen in Prozessmodellbibliotheken. Zu nennen sind beispielsweise Empfehlungsfunktionen während der Modellierung (Koschmider u. a. 2011) sowie die Erkennung von Duplikaten (Uba u. a. 2011) oder Varianten (Ekanayake u. a. 2012) von Geschäftsprozessmodellen. In Kapitel 5 wird ein Verfahren vorgeschlagen, um mit Hilfe eines Prozessmodells nach ähnlichen Prozessmodellen zu suchen. In Abbildung 1.4 ist diese Forschungsfrage im mittleren Teil illustriert. Dabei wird ein existierendes Prozessmodell als Eingabe verwendet, um ähnliche Modelle in einer Prozessmodellbibliothek zu finden und als Ergebnis auszugeben.

Zur Berechnung der Ähnlichkeit von Prozessmodellen werden typischerweise mehrere, verschiedene Dimensionen wie etwa die Beschriftungen von Modellelementen oder die Kontrollflussstruktur herangezogen (Schoknecht u. a. 2017b; Thaler u. a. 2017). In dieser Arbeit wird hingegen die Ansicht vertreten, dass insbesondere die Beschriftungen von Modellelementen für eine ähnlichkeitsbasierte Suchfunktion von Relevanz sind und andere Aspekte wie der Kontrollfluss eine geringere Wichtigkeit aufweisen. So spielt z. B. die Graphstruktur von Prozessmodellen beim Verschmelzen von Modellen eine größere Rolle als in einer Suchfunktion. Bei der Verschmelzung ist insbesondere ein wesentliches Ziel, dass nach der Verschmelzung ein Modell mit korrektem Kontrollfluss generiert wurde (La Rosa u. a. 2013), wodurch eine Ähnlichkeitsanalyse strukturelle Übereinstimmungen exakt identifizieren muss.

In einer Suchfunktion zur Modellierungsunterstützung ist die Struktur von Modellen hingegen nicht eminent wichtig. Ob etwa Aktivitäten durch eine AND- oder XOR-Verbindung verknüpft sind, ist für die Wiederverwendung nicht von besonderer Relevanz – die Verknüpfung könnte etwa leicht angepasst werden. Wichtiger erscheint es, dass inhaltlich ähnliche Modelle gefunden werden, die möglicherweise auch eine sehr unterschiedliche Kontrollflussstruktur beinhalten. Dabei liegt die Annahme zugrunde, dass sich die inhaltliche Bedeutung eines Prozessmodells im Sinne des menschlichen Verständnisses vor allem über textuelle Beschriftungen von Modellelementen erschließt,[4] während andere Aspekte wenig bzw. nichts dazu beitragen. Somit ergeben sich je nach Anwendungsfall jeweils andere Anforderungen an das zugrunde liegende Ähnlichkeitsmaß. Nützlich kann die Beschränkung auf textuelle Beschriftungen auch in Hinsicht auf die Korrektheit von gefundenen Modellen sein. Sollte ein gesuchtes Prozessmodell über einen fehlerhaften Kontrollfluss verfügen, so wird die in dieser Arbeit beschriebene Suchfunktion dadurch nicht beeinflusst, während es bei Ähnlichkeitsmaßen, die großen Wert auf den Kontrollfluss legen, zu Verzerrungen kommen könnte.

Zwar wurden in der wissenschaftlichen Literatur bereits eine ganze Reihe von Vorschlägen zur Berechnung der Ähnlichkeit von Geschäftsprozessmodellen für verschiedene Anwendungsfälle vorgestellt (siehe etwa die Übersichten in (Schoknecht u. a. 2017b) oder (Becker und Laue 2012)), die üblicherweise aber mehrere Dimensionen verwenden und zudem auf Prozessmodell-Matching basieren. Der in Kapitel 5 beschriebene Ansatz hingegen adaptiert ein Verfahren aus dem Bereich des Information Retrieval. Die eingesetzte Latent Semantic Analysis (LSA) (Deerwester u. a. 1990) beruht dabei auf einer mathematischen Darstellung der Beziehungen zwischen Dokumenten und den darin enthaltenen Termen in Form von Term-Dokument Matrizen, die letztlich eine Suche nach Dokumenten in einem Vektorraummodell gestatten. Im Folgenden wird diese Vektorraumrepräsentation dazu eingesetzt die Ähnlichkeit von Prozessmodellen basierend auf den textuellen Beschriftungen von Modellelementen zu bestimmen. Dabei wird auch die Schwierigkeit zur Berechnung eines Matchings umgangen, was im Vergleich mit fünf Matching-basierten Verfahren zu besseren Ergebnissen führt.

[4] Siehe dazu auch die Erläuterungen in (Leopold 2013, S. 10) zu einem Modellbeispiel mit und ohne textuelle Beschriftungen. Daraus wird ersichtlich, dass ohne textuelle Beschriftungen nur relativ abstrakte Informationen aus einem Prozessmodell abgeleitet werden können (etwa der Ablauf eines Prozesses), aber inhaltliche Angaben fehlen (etwa um welchen Prozess es sich handelt).

Neben den wissenschaftlichen Beiträgen stellt eine solche Suchmöglichkeit für die Praxis eine Option dar, existierende Modelle für die Modellierung neuer oder geänderter Prozesse wiederzuverwenden. Darüber hinaus könnte sie für weitere Anwendungsfälle, die eine ähnlichkeitsbasierte Suchfunktion benötigen oder durch sie unterstützt werden, eingesetzt werden. Dazu zählen z. B. die Erkennung von Prozessvarianten oder der Einsatz in einer Empfehlungsfunktion während der Modellierung, die aber nicht im Fokus dieser Arbeit stehen.

(iii) Wie kann eine Anfragesprache für Petri-Netz basierte Prozessmodelle realisiert werden?

Um nicht nur existierende Prozessmodelle für eine ähnlichkeitsbasierte Suche nutzen zu können, wird bezüglich der dritten Fragestellung eine Anfragesprache definiert, die eine größere Flexibilität bietet. Diese ist anwendbar für durch Petri-Netze repräsentierte Prozessmodelle und beschreibt letztlich ein (möglicherweise unvollständiges) Prozessmodell als Suchmuster zu dem ähnliche Modelle ermittelt werden sollen. Die Beschreibung dieser *Petri Net Query Model Language* (PNQML) beinhaltet eine Definition von Syntax, Semantik und Notation der Sprache, wie sie z. B. in (Fill und Karagiannis 2013) vorgeschlagen wird. Dargestellt ist diese Forschungsfrage im rechten Teil von Abbildung 1.4. Als Eingabe wird eine Anfrage in Form eines PNQML-Modells erwartet, sodass darauf basierend Modelle aus einer Prozessmodellbibliothek ausgegeben werden, die die in der Anfrage beschriebenen Restriktionen einhalten.

Es wird insbesondere auf die Ermöglichung von Query-by-Example fokussiert, da Modellierer nicht auf eine schwierige Verwendung von Datenbankanfragesprachen beschränkt sein sollen, um nach Modellen zu suchen. Daher wird eine Suchfunktionalität entworfen, die es ermöglicht, in der Anfragesprache modellierte Modellfragmente als Eingabe zu verwenden. Somit benötigen Modellierer zur Suche einerseits keine tiefergehenden Kenntnisse einer Datenbankanfragesprache[5] und müssen sich andererseits nicht auf eine textuelle Stichwortabfrage beschränken. Zudem wird erwartet, dass so auch komplexe Anfragen durch ein Modell relativ leicht ausgedrückt werden können. Durch eine Anlehnung an Petri-Netze bei der Definition der PNQML sollen Schwierigkeiten bei der praktischen Nutzung verringert und die Nützlichkeit

[5] Kenntnisse in Datenbankanfragesprachen können nicht vorausgesetzt werden. So weisen beispielsweise wenige der in (Harmon und Wolf 2011) angegebenen Berufsbezeichnungen von Prozessmodellierern auf einen verstärkten IT-Hintergrund hin. Ebenso wird von Cuff (1980) argumentiert, dass nicht zu erwarten sei, dass Nutzer von IT-Systemen Programmierkenntnisse besitzen werden.

einer Suchfunktion erhöht werden (vgl. dazu auch die Untersuchungen von verschiedenen Ansätzen zur Beschreibung von Datenbankanfragen in (Greene u. a. 1986)).

Darüber hinaus wird ein Algorithmus entworfen, um die Suchergebnisse einer Anfrage unter Berücksichtigung von Prozessmodell-Matching Verfahren zu berechnen und diese in einer Rangfolge anzuordnen. Somit werden nicht nur exakte Übereinstimmungen zwischen einem Anfragemodell und den Ergebnismodellen berücksichtigt, sondern es können auch ähnliche Modelle gefunden werden, die dennoch für einen Modellierer hilfreich sind. Dazu können beispielsweise die in Kapitel 4 beschriebenen Ansätze verwendet werden.

Bisherige verwandte Arbeiten auf diesem Gebiet fokussieren sich hauptsächlich auf BPMN-Modelle wie beispielsweise (Awad 2007) oder (Störrle und Acretoaie 2013), die dabei teilweise auch schon Prozessmodell-Matching Verfahren mit einbeziehen (z. B. (Awad u. a. 2008)). Hinsichtlich Petri-Netz basierter Prozessmodelle existiert zwar mit der PNQL ein Vorschlag von Xiao u. a. (2009), dieser berücksichtigt allerdings kein Matching, sodass nur Modelle in das Ergebnis einer PNQL-Anfrage aufgenommen werden, bei denen die Beschriftungen exakt mit denen der Anfrage übereinstimmen. Infolgedessen wird mit der PNQML keine grundsätzlich neue Idee beschrieben, sondern existierende Ansätze auf neuartige Weise miteinander kombiniert.

Aus praktischer Sicht erhöht sich durch eine solche Anfragesprache die Flexibilität bei der Suche nach Prozessmodellen im Vergleich zu einer ähnlichkeitsbasierten Suche (Forschungsfrage (ii)), indem kein existierendes Prozessmodell mehr vorliegen muss. Des Weiteren werden die syntaktischen Regeln von Petri-Netzen ergänzt um weitere Konstrukte, die es etwa ermöglichen nach Pfaden zwischen bestimmten Aktivitäten in Prozessmodellen zu suchen.

1.2 Publikationen im Rahmen der Dissertation

Während der Bearbeitung der im vorigen Teilkapitel beschriebenen Fragestellungen wurden mehrere Beiträge in wissenschaftlichen Konferenzen, Workshops und Journalen veröffentlicht. Diese erläutern ebenfalls Inhalte mit denen sich auch die vorliegende Arbeit befasst. In der folgenden Liste sind die entsprechenden Zitationsdaten der Veröffentlichungen aufsteigend nach Veröffentlichungsjahr angegeben:

- Fellmann, M., Koschmider, A., Schoknecht, A. (2014), **Analysis of Process Model Reuse Literature: Are Research Concepts Empirically Validated?**, in Hans-Georg Fill, Dimitris Karagiannis, Ulrich Reimer (Hrsg.), *Modellierung 2014*, Wien, Österreich, Gesellschaft für Informatik, S. 185–192.

- Koschmider, A., Fellmann, M., Schoknecht, A., Oberweis, A. (2014): **Analysis of process model reuse: Where are we now, where should we go from here?**, *Decision Support Systems* 66, S. 9–19.

- Cayoglu, U., Dijkman, R., Dumas, M., Fettke, P., García-Bañuelos, L., Hake, P., Klinkmüller, C., Leopold, H., Ludwig, A., Loos, P., Mendling, J., Oberweis, A., Schoknecht, A., Sheetrit, E., Thaler, T., Ullrich, M., Weber, I., Weidlich, M. (2014): **Report: The Process Model Matching Contest 2013**, in Niels Lohmann, Minseok Song, Petia Wohed, *Business Process Management Workshops*, Peking, China, Springer, S. 442–463.

- Antunes, G., Bakhshandeh, M., Borbinha, J., Cardoso, J., Dadashnia, S., Francescomarino, C. D., Dragoni, M., Fettke, P., Gal, A., Ghidini, C., Hake, P., Khiat, A., Klinkmüller, C., Kuss, E., Leopold, H., Loos, P., Meilicke, C., Niesen, T., Pesquita, C., Péus, T., Schoknecht, A., Sheetrit, E., Sonntag, A., Stuckenschmidt, H., Thaler, T., Weber, I., Weidlich, M. (2015): **The Process Model Matching Contest 2015**, in Jens Kolb, Henrik Leopold, Jan Mendling (Hrsg.), *6th International Workshop on Enterprise Modelling and Information Systems Architectures (EMISA 2015)*, Innsbruck, Österreich, Gesellschaft für Informatik, S. 127–155.

- Koschmider, A., Figl, K., Schoknecht, A. (2015): **A Comprehensive Overview of Visual Design of Process Model Element Labels**, in Manfred Reichert, Hajo Reijers (Hrsg.), *Business Process Management Workshops*, Innsbruck, Österreich, Springer, S. 571–582.

- Thaler, T., Schoknecht, A., Fettke, P., Oberweis, A., Laue, R. (2016): **A Comparative Analysis of Business Process Model Similarity Measures**, in Marlon Dumas, Marcelo Fantinato (Hrsg.), *Business Process Management Workshops*, Rio de Janeiro, Brasilien, Springer, S. 310–322.

- Schoknecht, A., Fischer, N., Oberweis, A. (2016): **Process Model Search Using Latent Semantic Analysis**, in Marlon Dumas, Marcelo Fantinato (Hrsg.), *Business Process Management Workshops*, Rio de Janeiro, Brasilien, Springer, S. 283–295.

- Schoknecht, A., Thaler, T., Fettke, P., Oberweis, A., Laue, R. (2017): **Similarity of Business Process Models — A State-of-the-Art Analysis**, *ACM Computing Surveys*, 50(4), S. 52:1–52:33.

- Schoknecht, A., Oberweis, A. (2017): **LS3: Latent Semantic Analysis-based Similarity Search for Process Models**, *EMISA Journal*, 12(2), S. 1–22.

1.3 Strukturierung der Inhalte

In Abbildung 1.5 ist die weitere Struktur der Arbeit dargestellt. Die Kapitel 2 und 3 beschreiben Grundlagen zum Verständnis der weiteren Arbeit. Dabei wird zunächst eine Einführung in das Management von Geschäftsprozessen gegeben, wobei insbesondere auf die Modellierung von Prozessen durch Petri-Netze fokussiert wird. Zudem wird vertieft auf die Ähnlichkeit von Prozessmodellen eingegangen, was den zentralen Aspekt der entwickelten Suchfunktionen darstellt. In Kapitel 3 werden anschließend grundlegende Techniken zur automatisierten Verarbeitung natürlicher Sprache vorgestellt. Des Weiteren werden Verfahren zur Berechnung der Ähnlichkeit von Wörtern erläutert sowie die Latent Semantic Analysis detailliert vorgestellt. Schließlich werden zum Abschluss dieses Kapitels noch verschiedene Evaluationsmaße aus dem Information Retrieval eingeführt, die zur Bewertung der entwickelten Suchverfahren eingesetzt werden.

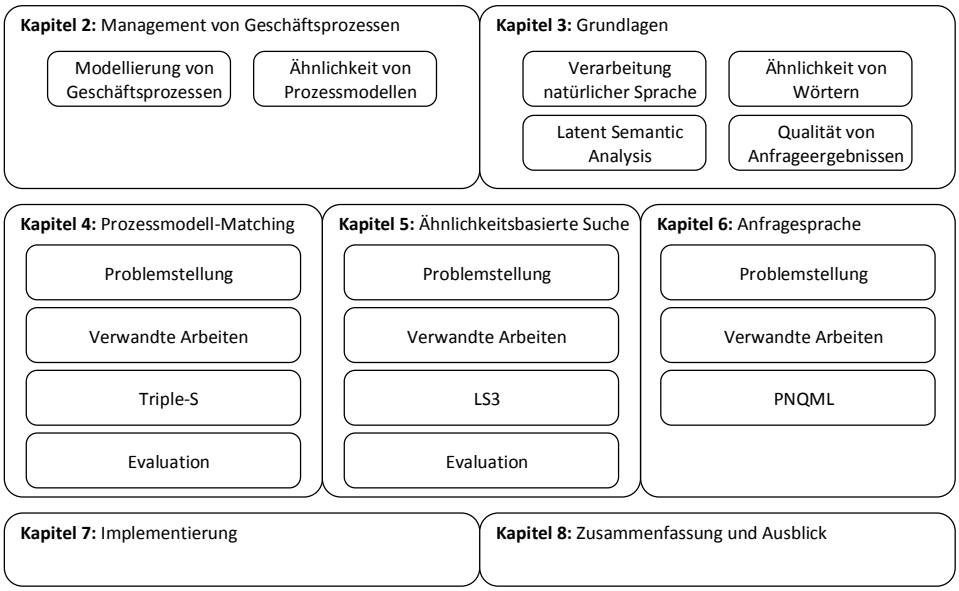


Abb. 1.5: Übersicht über den Aufbau der Arbeit

Nach diesem Grundlagenteil stellen die Kapitel 4 bis 6 den eigentlichen Kern der Arbeit dar. Jedes Kapitel widmet sich dabei einer der zuvor erläuterten Forschungsfragen. In Kapitel 4 werden zunächst die entwickelten Prozessmodell-Matching An-

sätze beschrieben, um in Kapitel 5 auf die ähnlichkeitsbasierte Suche einzugehen. Die dritte Forschungsfrage wird anschließend durch die Vorstellung der Petri Net Query Model Language in Kapitel 6 behandelt. Alle drei Kapitel sind grundsätzlich gleich aufgebaut und beinhalten zunächst eine Beschreibung der zugrunde liegenden Problemstellung sowie eine Betrachtung relevanter verwandter Arbeiten. Anschließend werden die jeweiligen Suchverfahren vorgestellt: *Triple-S* für das Prozessmodell-Matching, *LS3* zur ähnlichkeitsbasierten Suche und die *PNQML* als Anfragesprache für Petri-Netz basierte Prozessmodelle.

Daraufhin wird in Kapitel 7 noch auf eine mögliche Integration der Suchverfahren mit weiteren Komponenten zur Modellierung und Speicherung von Prozessmodellen eingegangen. Dazu wird eine mögliche Architektur eines solchen Systems vorgestellt sowie beispielhaft eine prototypische Implementierung beschrieben. Schließlich enthält Kapitel 8 eine Zusammenfassung der Erkenntnisse und einen Ausblick auf zukünftige Forschungs- und Entwicklungsmöglichkeiten.

Kapitel 2

Management von Geschäftsprozessen

Das Management von Geschäftsprozessen ist ein zentraler Aspekt heutiger Unternehmensführung, der es Unternehmen erlaubt, flexibel auf sich ändernde Bedingungen zu reagieren (Kohlbacher 2010; Harmon und Wolf 2014). Prozessmodelle werden dabei für zahlreiche Aspekte eingesetzt, wodurch Sammlungen mit hunderten von Modellen entstehen können. Um solche Sammlungen effektiv verwalten und das darin enthaltene Wissen wiederverwenden zu können, greifen viele fortschrittliche Funktionalitäten zur Unterstützung der Prozessmodellierung und Verwaltung von Prozessmodellen auf ein Ähnlichkeitsmaß zwischen Prozessmodellen oder Aktivitäten in Modellen zurück (z. B. Koschmider u. a. (2011) in einer Recommender-Funktionalität für Prozessmodellierungswerkzeuge oder Uba u. a. (2011) zur Erkennung von Duplikaten in Prozessmodellsammlungen).

Daher gibt dieses Kapitel einen Überblick über das Geschäftsprozessmanagement sowie die Modellierung von Geschäftsprozessen. Zunächst wird auf das Management von Geschäftsprozessen eingegangen, um ein Verständnis dafür zu schaffen, was unter einem Geschäftsprozess zu verstehen ist und in welchen Bereichen Prozessmodelle Verwendung finden. In Kapitel 2.1 wird anschließend detaillierter auf Prozessmodelle und die zugrunde liegende Modellierung eingegangen. Dabei werden Petri-Netze als eine Möglichkeit zur Repräsentation von Geschäftsprozessen sowie ein verbreitetes XML-Austauschformat als technische Darstellungsform vorgestellt. Zudem erfolgt in Kapitel 2.1.3 eine Beschreibung existierender Ansätze zur Speicherung von Modellen in Prozessmodellbibliotheken, da die in Kapitel 6.3 beschriebene Anfragesprache für

eine solche Bibliothek zur Berechnung von Anfragen definiert wird. Der für diese Arbeit zentrale Ähnlichkeitsaspekt von Prozessmodellen wird in Kapitel 2.2 ausgeführt. Dabei wird der Begriff Ähnlichkeit im Kontext von Prozessmodellen erläutert sowie auf die Quantifizierung von Ähnlichkeit eingegangen.

Die Ausrichtung von Unternehmen auf eine prozessorientierte Sichtweise lässt sich bis zu den Anfängen des 20. Jahrhunderts zurückführen. Die Entwicklung des Fließbands durch Ford bzw. die Argumentation von Taylor (1911, S. 68-70) im Sinne einer systematischen Analyse von Arbeit, um eine bestmögliche Art und Weise der Ausführung von Aufgaben zu identifizieren, lassen sich etwa als erste Schritte hin zu einer Prozessorientierung von Unternehmen auffassen. Porter (1985) führte diese Gedanken durch die Idee einer Wertschöpfungskette, in der Aktivitäten aneinander gereiht werden, um ein Produkt für einen Markt herzustellen, weiter. Darüber hinaus legten Davenport (1993) durch sein Konzept der Prozessinnovation sowie Hammer und Champy (1994) durch die Einführung des Prozess-Reengineerings weitere Grundlagen für das wirtschaftswissenschaftliche Verständnis des Geschäftsprozessmanagements. Aus technischer Sicht spielen für das Verständnis von Geschäftsprozessmanagement Konzepte aus den 70er Jahren zu Büroinformationssystemen sowie die Entwicklung von Workflow-Systemen in den neunziger Jahren eine große Rolle. Dies resultierte darin, dass sich heutzutage in vielen Informationssystemen wie beispielsweise Enterprise Resource Planning Systemen prozessunterstützende Komponenten finden (van der Aalst 2013).

In dieser Arbeit werden die technische Sichtweise und eine Problemstellung aus diesem Bereich in den Fokus gestellt. Daher wird auch zum grundlegenden Verständnis eines *Geschäftsprozesses* in Definition 2.1 auf eine Beschreibung von Oberweis (1996) zurückgegriffen. In der weiteren Arbeit werden die Begriffe Prozess und Geschäftsprozess synonym zueinander verwendet.

Definition 2.1: Geschäftsprozess

Ein Geschäftsprozess ist eine „Menge von manuellen, teilautomatisierten oder automatisierten Aktivitäten, die in einem Betrieb nach bestimmten Regeln auf ein bestimmtes Ziel hin ausgeführt werden. Die Aktivitäten hängen über betroffene Personen, Maschinen, Dokumente, Betriebsmittel und ähnliches miteinander zusammen“ (Oberweis 1996).

Ein Geschäftsprozess besteht demnach aus Aktivitäten, die in einer bestimmten Reihenfolgenbeziehung zueinander stehen.[1] Die Aktivitäten selbst können durch Personen (manuell) bzw. unterstützt durch oder vollständig automatisiert von Informationssystemen ausgeführt werden. Das Geschäftsprozessmanagement befasst sich schließlich mit der Gestaltung und Verwaltung von Prozessen. Der Definition von Weske (2012) folgend, kann *Geschäftsprozessmanagement* definiert werden als:

> **Definition 2.2: Geschäftsprozessmanagement**
>
> Geschäftsprozessmanagement umfasst Konzepte, Methoden und Techniken, um den Entwurf sowie die Verwaltung, Konfiguration, Durchführung und Analyse von Geschäftsprozessen zu unterstützen (Weske 2012) (eigene Übersetzung).

Um Tätigkeiten innerhalb des Geschäftsprozessmanagements strukturiert ausführen zu können, hat sich ein sogenannter *Lebenszyklus* für Geschäftsprozesse etabliert (siehe z. B. (Dumas u. a. 2013b), (Weske 2012) oder (van der Aalst 2013)). Abbildung 2.1 zeigt einen Lebenszyklus nach (Dumas u. a. 2013b). Anhand der folgenden Beschreibung der Phasen dieses Lebenszyklus wird erkenntlich, wie zentral Prozessmodelle für das Geschäftsprozessmanagement sind.

- *Prozessidentifikation:* Die erste Phase des Lebenszyklus befasst sich mit der Identifikation von Prozessen, die für eine bestimmte Fragestellung von Relevanz sind. Dabei kann es sich z. B. um die Frage handeln, welche Prozesse für Mitarbeiter als Dokumentation modelliert werden sollen oder ob die Durchlaufzeit durch eine Neugestaltung eines Prozesses verkürzt werden kann. Diese Phase bildet die Grundlage für die anschließend zyklisch ablaufenden Tätigkeiten.
- *Prozessmodellierung:* Während dieser Phase werden die Prozesse modelliert, sodass der aktuelle Arbeitsablauf dargestellt wird. Dies geschieht meist durch die Verwendung einer grafischen Modellierungssprache, was in der Erstellung eines Diagramms eines Prozesses resultiert. Auf diesen Aspekt der Modellerstellung wird in Kapitel 2.1 vertieft eingegangen, da sich diese Arbeit vorrangig in diese Phase des Lebenszyklus einordnen lässt. Bei der Prozessmodellierung

[1] Davenport (1993) bezeichnet dies als spezifische Ordnungsbeziehung zwischen Aktivitäten.

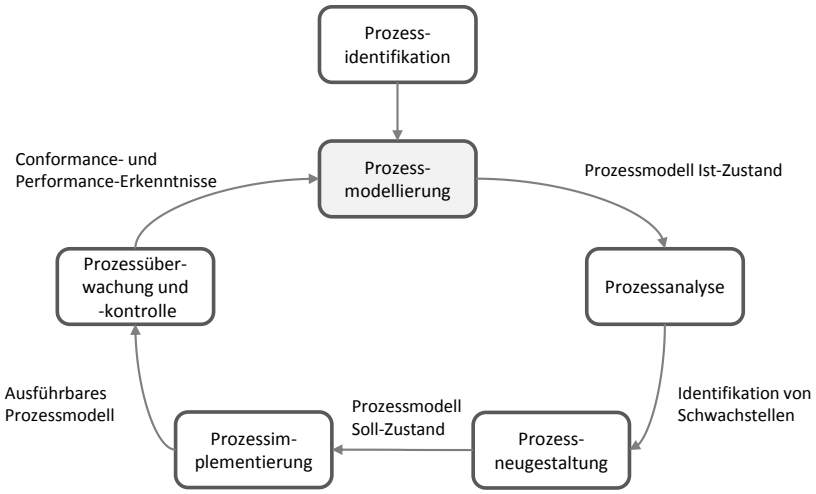


Abb. 2.1: Lebenszyklus von Geschäftsprozessen (Darstellung in Anlehnung an (Dumas u. a. 2013b))

können beispielsweise die entwickelten Suchfunktionalitäten unterstützend eingesetzt werden, um ähnliche Modellteile zu finden und diese wiederzuverwenden. Dadurch muss ein Modell nicht von Grund auf neu modelliert werden, sondern ein Modellierer kann auf bereits bestehende Modelle zurückgreifen.

- *Prozessanalyse:* In der dritten Phase finden analytische Aktivitäten statt, um Verbesserungsmöglichkeiten und Schwachstellen in Prozessen zu identifizieren. Zum Beispiel kann quantitativ untersucht werden, wie hoch die Durchlaufzeit momentan ist und wie diese potentiell verkürzt werden kann. Solche Analysetätigkeiten werden häufig anhand von Modellen durchgeführt. So kann etwa die Durchlaufzeit eines Prozesses anhand von Prozesssimulationen auf Basis eines Modells bestimmt werden. Auch bei diesem Aspekt können Suchmöglichkeiten hilfreich sein, um etwa ähnliche Modellteile zu finden und diese konsistent umzugestalten oder in ein neues Modell auslagern zu können.

- *Prozessneugestaltung:* Um die Verbesserungsmöglichkeiten aus der Analysephase umzusetzen, werden anschließend denkbare Optionen zur Neugestaltung eines Prozesses in Betracht gezogen. Diese können etwa zu Änderungen am Ablauf eines Prozesses führen, was in einem geänderten Soll-Prozess resultiert. Dieser wird üblicherweise wiederum durch ein Modell dargestellt, um

beispielsweise eine Simulation des geänderten Prozesses vor der eigentlichen Implementierung durchführen zu können.

- *Prozessimplementierung:* In der fünften Phase wird daraufhin der geänderte Prozess realisiert. D. h. es werden möglicherweise neue Informationssysteme entwickelt oder eingeführt, die den neuen Prozess unterstützen, existierende Informationssysteme angepasst und die Mitarbeiter im Rahmen eines organisatorischen Änderungsmanagements mit den Anpassungen vertraut gemacht. Kurz gefasst werden alle notwendigen Aktivitäten durchgeführt, um den geänderten Prozess im Unternehmen einzuführen. Bei (teil-)automatisierten Prozessen werden dazu Prozessmodelle in ein sogenanntes *ausführbares Prozessmodell* transformiert, sodass sie durch entsprechende Informationssysteme unterstützt und ausgeführt werden können.
- *Prozessüberwachung und -kontrolle:* Um überprüfen zu können, ob die Änderungen auch wirklich die gewünschten Effekte erzielen, muss die Prozessausführung kontinuierlich überwacht und kontrolliert werden. Die Erkenntnisse dieser Phase fließen anschließend wieder in die Modellierungs-, Analyse- und Neugestaltungsphasen ein, sodass eine kontinuierliche Verbesserung von Prozessen möglich wird und sich schließlich ein Kreislauf ergibt.

Aus dieser Beschreibung wird ersichtlich, welche zentrale Rolle Prozessmodelle im Geschäftsprozessmanagement einnehmen. Sie sind entweder ein wichtiges Ergebnis verschiedener Phasen (Prozessmodellierung und -neugestaltung) oder ein essentieller Teil von Phasen (Prozessanalyse und Implementierung) des Lebenszyklus. Im Folgenden wird darauf eingegangen, was durch Prozessmodelle ausgedrückt wird und wie sie sich durch Petri-Netze repräsentieren lassen. Insbesondere wird beschrieben, welche Aspekte von Prozessmodellen zur Berechnung der Ähnlichkeit genutzt werden können und für welche Zielsetzungen dies sinnvoll erscheint.

2.1 Modellierung von Geschäftsprozessen

Zur Beschreibung und Definition von Geschäftsprozessmodellen ist zunächst zu klären, worum es sich bei einem Modell handelt. Nach Stachowiak (1973, S. 131–132) muss ein Modell die folgenden Merkmale erfüllen:

- **Abbildungsmerkmal:** Modelle sind Abbildungen (Repräsentationen) von Originalen aus der realen Welt bzw. von künstlichen Originalen, wobei jeweils der mathematische Abbildungsbegriff zugrunde liegt.
- **Verkürzungsmerkmal:** Modelle erfassen nicht alle Aspekte bzw. Attribute eines Originals, sondern beschränken sich auf die für den Modellierer relevanten Aspekte bzw. Attribute.
- **Pragmatisches Merkmal:** Modelle sind nicht eindeutig, sondern werden von Modellierern zu einem bestimmten Zeitpunkt für bestimmte Modellnutzer und bezüglich eines bestimmten Zwecks entworfen.

Die Merkmalsaufzählung von Stachowiak (1973) abstrahiert von dem Einfluss, den die subjektive Wahrnehmung eines Modellierers auf das letztendliche Modell ausübt. Schuette und Rotthowe (1998) argumentieren daher, dass für das Zutreffen des Abbildungsmerkmals die Realität eine epistomologische Objektivität aufweisen und damit unabhängig von einem Modellierer sein müsste. Sollte dies zutreffen, müsste die subjektive Wahrnehmung eines Modellierers exakt mit der Realität übereinstimmen, was nicht sonderlich realistisch erscheint. So beschreibt etwa Norman (1983), dass die Sicht der Welt einer Person auf den Konzeptualisierungen und dem Wissen dieser Person basiert. Personen bilden daher *mentale Modelle* zum Verständnis ihrer Umgebung, wobei diese u. a. wiederum instabil und unvollständig sind und zudem keine wohldefinierten Grenzen besitzen. Damit können sie fehlerhafte, sich widersprechende oder überflüssige Konzepte beinhalten (Norman 1983). Somit wird im Folgenden auch von dem formalen Abbildungsmerkmal nach Stachowiak (1973) abgesehen und die Sichtweise angenommen, dass ein Modell das Ergebnis einer Modellierungstätigkeit von Modellierern ist (Schuette und Rotthowe 1998).

Definition 2.3: Geschäftsprozessmodell

Ein Geschäftsprozessmodell (Prozessmodell) ist eine durch einen oder mehrere Modellierer erstellte Abbildung eines existierenden oder zu entwerfenden Geschäftsprozesses. In einem Geschäftsprozessmodell können u. a. die in dem abgebildeten Geschäftsprozess ausgeführten Aktivitäten, beteiligte Rollen und Ressourcen, Ereignisse und Ausführungsbedingungen repräsentiert sein.

Zur Modellierung stehen verschiedene informale, semi-formale und formale Repräsentationsmöglichkeiten zur Auswahl (Desel und Juhás 2001), wobei die im Weiteren betrachteten Modelle semi-formale oder formale Charakteristiken aufweisen und typischerweise als Ereignisgesteuerte Prozessketten (EPK) (Keller u. a. 1992), Business Process Model and Notation Diagramme (*Business Process Model and Notation (BPMN)* 2011) oder Petri-Netze (Murata 1989) repräsentiert sind. Daraus wird der Fokus auf *imperative Prozessmodelle* deutlich, in denen die Ausführungsreihenfolge von Aktivitäten explizit angegeben ist gegenüber *deklarativen Prozessmodellen*, in denen Bedingungen an die Ausführungsreihenfolge gestellt werden (Fahland u. a. 2009). Da deklarative Modelle teilweise in imperative Prozessmodelle transformiert werden können (Prescher u. a. 2014) und insbesondere weitere Herausforderungen an die Ähnlichkeitsbestimmung von Prozessmodellen stellen würden, werden diese im Weiteren nicht betrachtet.

Hinsichtlich der Ausführung eines Prozesses wird der Begriff *Prozessinstanz* verwendet, wobei eine Instanz in der realen Welt beobachtet oder beispielsweise anhand eines Modells durch Simulation erzeugt werden kann. In dem Szenario der Beantragung einer Dienstreise in Kapitel 1 (siehe auch Abbildung 1.1) entspricht dies einem Durchlauf des abgebildeten Prozesses. Beispielsweise füllt ein Mitarbeiter einen Antrag aus und gibt ihn ab, anschließend wird der Antrag genehmigt und an den Mitarbeiter zurückgegeben. Eine Möglichkeit zur Beschreibung solcher Prozessinstanzen ist ihre Darstellung als *Trace*, welche die ausgeführten Aktivitäten beinhaltet. Für eine formale Definition von Prozessinstanzen zu Petri-Netz basierten Prozessmodellen siehe Definition 2.5.

2.1.1 Petri-Netze zur Repräsentation von Geschäftsprozessen

Petri-Netze wurden ursprünglich von Carl Adam Petri in seiner Dissertation entworfen (Petri 1962). Sie stellen eine graphische und mathematisches Modellierungssprache zur Modellierung von Systemen zur Informationsverarbeitung dar, die auch zur Simulation des Systemverhaltens eingesetzt werden kann (Murata 1989). Darüber hinaus wurden sie auch für viele weitere Anwendungsgebiete verwendet (siehe etwa die Aufzählung in (Murata 1989)), wobei sie im Folgenden zur Darstellung von Geschäftsprozessmodellen dienen.

Petri-Netze sind spezielle Arten von gerichteten, gewichteten, bipartiten Graphen, die aus zwei Arten von Knoten bestehen. *Stellen* repräsentieren dabei statische Modellele-

mente und im Kontext von Prozessen beispielsweise Bedingungen oder Ressourcen. *Transitionen* stellen aktive Modellelemente dar und werden im Prozesskontext mit Aktivitäten, Aktionen oder Ereignissen assoziiert. Kanten in einem solchen Graphen verbinden Stellen mit Transitionen und Transitionen mit Stellen, also nur Knoten jeweils unterschiedlichen Typs. Darüber hinaus existieren sogenannte Marken, die den Zustand eines Petri-Netzes (die Markierung) darstellen, indem sie Stellen zugewiesen werden. Im Zusammenhang mit Geschäftsprozessen kann eine Marke in einer Stelle z. B. das Vorhandensein einer Ressource symbolisieren. Eine formale Definition eines Petri-Netzes ist im Folgenden beschrieben:

Definition 2.4: Petri-Netz

Ein Petri-Netz ist ein 5-Tupel $P = (S, T, F, W, M_0)$, wobei

- $S = \{s_1, s_2, \cdots, s_m\}$ eine endliche Menge von Stellen und
- $T = \{t_1, t_2, \cdots, t_n\}$ eine endliche Menge von Transitionen sind,
- $F \subseteq (S \times T) \cup (T \times S)$ eine Menge von Kanten (die Flussrelation) ist,
- $W : F \to \{1, 2, 3, \cdots\}$ eine Gewichtungsfunktion und
- $M_0 : S \to \{0, 1, 2, 3, \cdots\}$ eine Anfangsmarkierung darstellen und
- $S \cap T = \emptyset$ sowie $S \cup T \neq \emptyset$ gilt (Murata 1989).

Ein Beispiel eines Petri-Netzes ist zur Verdeutlichung in Abbildung 2.2 dargestellt. Dieses enthält die Menge $S = \{s_1, s_2, s_3, s_4, s_5\}$ an Stellen (abgebildet als Kreise) sowie die Menge $T = \{t_1, t_2, t_3, t_4, t_5\}$ an Transitionen (abgebildet als Rechtecke). Die Flussrelation F ist über gerichtete Kanten dargestellt, wobei beispielhaft zwei Gewichte der Gewichtungsfunktion W jeweils durch eine 1 angegeben sind. Da die Gewichte im Weiteren keine Bedeutung für die entwickelten Suchfunktionalitäten und die Ähnlichkeitsberechnungen haben, wird für alle Kantengewichte der Wert 1 festgelegt, wodurch sich ein sogenanntes *gewöhnliches Petri-Netz* ergibt (Murata 1989).

Marken sind durch ausgefüllte Kreise dargestellt, sodass die Anfangsmarkierung M_0 des Beispiel-Netzes in Abbildung 2.2 durch den Vektor $(1, 0, 0, 0, 0)$ der Stellen gegeben ist. Da auch das Ablaufverhalten eines Prozessmodells für die weitere Arbeit nicht

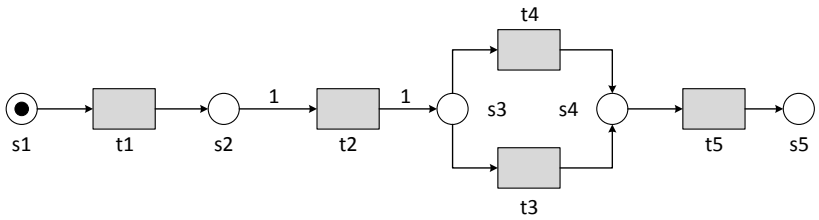


Abb. 2.2: Beispiel eines Petri-Netzes

berücksichtigt wird, enthalten die folgenden Petri-Netze keine Marken. Bezüglich der Ausführung einer Prozessinstanz wird folgende Definition verwendet:

Definition 2.5: Trace, Tracelänge

Eine Trace σ eines Petri-Netzes P ist eine Folge von Aktivitäten (Transitionen) $t \in T$, die die Reihenfolge der ausgeführten Aktivitäten angibt. Sie wird mit $\sigma =< t_1, \cdots, t_i, \cdots, t_n >$ beschrieben, wobei t_i identisch zu t_j sein kann ($i \neq j$), da eine Aktivität mehrfach durchgeführt werden kann. Die Länge einer Trace σ ist die Anzahl der darin enthaltenen Aktivitäten (Schoknecht u. a. 2017b).

Für die Suchfunktionalitäten sind insbesondere die *Beschriftungen* von Modellelementen von Bedeutung, da diese die Semantik eines Modells wiedergeben, indem sie etwa beschreiben, welche Aufgaben in einer Aktivität durchgeführt werden (siehe dazu auch die Erläuterungen in (Leopold 2013, S. 10)). Sei $\mathfrak{L}$ eine Menge von Beschriftungen (Label) und $N = S \cup T$ die Menge aller Knoten eines Petri-Netzes. Dann weist die Funktion $\ell : N \rightarrow \mathfrak{L}$ jedem Knoten eine Beschriftung zu. Beispiele für Beschriftungen in Abbildung 2.2 sind die Stellenbezeichnungen $s_1, \cdots, s_5$ sowie die Transitionsbeschriftungen $t_1, \cdots, t_5$.

Im Kontext von Geschäftsprozessen, die durch Petri-Netze repräsentiert sind, erläutern die Beschriftungen von Stellen etwa, welche Bedingungen vorliegen bzw. die Beschriftungen von Transitionen, welche Aktivitäten ausgeführt werden. Zudem können auch an einem Prozess beteiligte Rollen oder Geschäftsobjekte in diesen Beschriftungen erwähnt sein. Leopold u. a. (2012) identifizieren etwa für Aktivitäten drei übliche Bestandteile von Beschriftungen: eine auszuführende Aktion, ein Geschäftsobjekt

auf dem die Aktion ausgeführt wird und ein optionales Fragment mit weiteren Angaben. Dabei bestehen Label typischerweise nur aus wenigen Wörtern oder kurzen Sätzen, wobei eine ideale Beschriftungslänge für eine gute visuelle Gestaltung eines Modells noch eine offene Forschungsfrage darstellt (Koschmider u. a. 2015). Auch für die Konstruktion von Beschriftungen von Modellelementen gibt es Vorschläge wie etwa die Verwendung von Richtlinien zur Beschriftung (Mendling u. a. 2010) oder die Verwendung von Satzschablonen (Caporale 2016). Die einheitliche Verwendung solcher Vorschläge erscheint jedoch in der Praxis schwer durchsetzbar, wie sich beispielsweise bei der Untersuchung eines Modelldatensatzes hinsichtlich der verwendeten Richtlinien zur Beschriftung zeigte. Dabei wurde ein Beschriftungsstil für 60 % der Aktivitäten verwendet, ein weiterer für 34 % und die restlichen 6 % der Aktivitäten wurden wiederum auf andere Weise beschriftet (Mendling u. a. 2010).

Weiterhin relevant für die weitere Arbeit sind der *Vor-* und *Nachbereich* eines Knotens $n \in N$. Dabei liegen im Vorbereich von n alle die Knoten n_i, die folgende Bedingung erfüllen: $\bullet n = \{n_i \mid (n_i, n) \in F\}$. Im Nachbereich hingegen sind alle die Knoten enthalten, die folgende Bedingung erfüllen: $n\bullet = \{n_i \mid (n, n_i) \in F\}$. In Abbildung 2.2 besteht etwa der Vorbereich des Knotens s_3 aus der Transition t_2, während der Nachbereich die Transitionen $\{t_3, t_4\}$ enthält. Ist der Vorbereich einer Stelle leer, so wird sie als *Quelle* bezeichnet; ist der Nachbereich einer Stelle leer, wird sie *Senke* genannt. In dem Beispielnetz stellen s_1 eine Quelle und s_5 eine Senke dar. Zudem wird ein Petri-Netz als *stark verbunden* bezeichnet, wenn von jedem Knoten aus jeder andere Knoten über einen gerichteten Pfad erreicht werden kann.

Ein *Pfad* zwischen zwei Knoten $a \in N$ und $b \in N$ ist dabei eine Folge von Knoten $n_1, \cdots, n_k \in N$ mit $a = n_1$ und $b = n_k$, sodass für alle $i \in 1, \cdots, k-1$ eine entsprechende Kante $(n_i, n_{i+1}) \in F$ existiert. Dargestellt wird ein solcher Pfad als $a \hookrightarrow b$. Die Länge eines Pfads wird mit $len(a \hookrightarrow b)$ bezeichnet und ergibt sich als die Anzahl an Kanten, über die a und b in diesem Pfad verbunden sind. Im Beispiel aus Abbildung 2.2 enthält der einzige Pfad von s_1 nach t_2 die Kanten $(s_1, t_1), (t_1, s_2), (s_2, t_2)$ und seine Länge beträgt 3.

Weitere Details zu Petri-Netzen und den zuvor erläuterten Aspekten finden sich in (Murata 1989; Reisig 1985). Neben den bereits in Definition 2.4 beschriebenen Bestandteilen eines Petri-Netzes müssen diese für die vorliegende Arbeit noch die Bedingungen von *Workflow-Netzen* nach van der Aalst (1998) erfüllen:

Definition 2.6: Workflow-Netz

Ein Petri-Netz $P = (S, T, F, W, M_0)$ ist genau dann ein Workflow-Netz, wenn es folgende Bedingungen erfüllt:

- P enthält zwei spezielle Stellen i und o für die gilt, dass i eine Quelle ($\bullet i = \emptyset$) und dass o eine Senke ($o\bullet = \emptyset$) darstellt.
- Wenn eine Transition t^* zu P hinzugefügt wird, die o mit i über zwei Kanten $(o, t^*) \in F$ und $(t^*, i) \in F$ verbindet, so ist das resultierende Petri-Netz stark verbunden.

Diese Einschränkung auf Workflow-Netze ist notwendig, um den in Kapitel 4.3.1 beschriebenen Ansatz für das Prozessmodell-Matching eindeutig berechnen zu können. Sie stellt allerdings nur eine relativ geringe Restriktion dar, da in einigen Fällen durch einfaches Hinzufügen von Transitionen und Stellen aus einem Petri-Netz ein entsprechendes Workflow-Netz erzeugt werden kann.[2] Dies ist schematisch in Abbildung 2.3 dargestellt. Sollte ein Petri-Netz mehrere Stellen enthalten, die den Beginn eines Prozesses darstellen, so werden diese mit einer neuen Transition t_{in} sowie einer neuen Stelle i verbunden, sodass eine Quelle vorliegt. Entsprechendes wird durchgeführt, sollte ein Petri-Netz mehrere Stellen enthalten, die das Ende eines Prozesses repräsentieren. Dann werden dem Netz eine Stelle o und eine Transition t_{out} hinzugefügt, sodass sich der rechte Teil von Abbildung 2.3 ergibt.

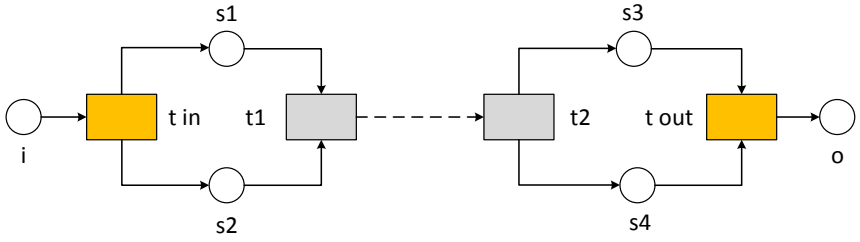


Abb. 2.3: Schematische Darstellung zur Konstruktion von Workflow-Netzen. Die goldenen Transitionen sowie die Stellen i und o repräsentieren die zu ergänzenden Modellelemente.

[2] Dieses Vorgehen ist im Wesentlichen auch in (Polyvyanyy u. a. 2012) für BPMN- und EPK-Modelle beschrieben.

Sollten zudem Transitionen mit leerem Vorbereich existieren, so wird jeweils eine Kante von Stelle i zu diesen Transitionen hinzugefügt. Analog wird bei Transitionen mit leerem Nachbereich jeweils eine Kante zu der Stelle o hinzugefügt. Zusammenfassend ergibt sich folgende Definition für die in dieser Arbeit verwendete Petri-Netz Repräsentation von Prozessmodellen:

Definition 2.7: Beschriftetes Workflow-Netz

Ein beschriftetes Workflow-Netz ist ein 4-Tupel $BWN = (S, T, F, \ell)$ mit

- $S = \{s_1, s_2, \cdots, s_m\}$: endliche Menge von Stellen,
- $T = \{t_1, t_2, \cdots, t_n\}$: endliche Menge von Transitionen,
- $F \subseteq (S \times T) \cup (T \times S)$: Menge von Kanten (Gewicht jeweils 1) und
- $\ell : N \to \mathfrak{L}$: einer Beschriftungsfunktion mit $N = S \cup T$.

Des Weiteren muss $S \cap T = \emptyset$, $\mid S \mid \geq 2$ und $\mid T \mid \geq 1$ gelten, sodass ein beschriftetes Workflow-Netz mindestens zwei Stellen (Quelle und Senke) sowie eine Aktivität (Transition) enthält. Zudem muss ein beschriftetes Workflow-Netz die Bedingungen an Workflow-Netze nach Definition 2.6 erfüllen.

2.1.2 Petri Net Markup Language

Petri-Netze können mit Hilfe der *Petri Net Markup Language* (PNML) (Weber und Kindler 2003), einer XML-Sprache, repräsentiert werden, um sie so auch zwischen verschiedenen Modellierungswerkzeugen austauschen zu können. Diese XML-Sprache wird auch im Folgenden genutzt und Listing 2.1 zeigt ein unvollständiges Beispiel einer PNML-Datei, das den Anfang des in Abbildung 2.2 dargestellten Modells wiedergibt. Dieses Listing zeigt zudem, dass für die Ähnlichkeitsberechnung und Suchfunktionalitäten relevante Informationen wie die Beschriftungen von Modellelementen oder der Kontrollfluss aus den PNML-Dateien entnommen werden können. So sind etwa die Beschriftungen von Modellelementen in den <*text*>-Tags enthalten und der Kontrollfluss lässt sich aus den Quell- und Zielknoten von <*arc*>-Elementen ableiten. Zur eindeutigen Identifikation von Elementen innerhalb eines Modells werden jeweils ID-Attribute verwendet. Zudem enthalten PNML-Dateien

auch grafische Informationen zur Darstellung in Modellierungswerkzeugen in den <*graphics*>-Elementen.

Listing 2.1: Beispiel einer PNML-Datei

```
<pnml>
  <net id="exampleModel">
    <place id="p1">
      <name>
        <text>s1</text>
        <graphics><offset x="50" y="110"/></graphics>
      </name>
      <graphics>
        <position x="50" y="70"/>
        <dimension x="40" y="40"/>
      </graphics>
      <initialMarking>
        <text>1</text>
      </initialMarking>
    </place>
    <transition id="t1">
      <name>
        <text>t1</text>
        <graphics><offset x="120" y="110"/></graphics>
      </name>
      <graphics>
        <position x="125" y="70"/>
        <dimension x="40" y="40"/>
      </graphics>
    </transition>
    <arc id="a1" source="p1" target="t1"></arc>
    ...
  </net>
</pnml>
```

2.1.3 Speicherung von Prozessmodellen in Modellbibliotheken

Neben der Speicherung als Dateien werden Prozessmodelle auch in sogenannten Prozessmodell-Repositorien oder -Bibliotheken gespeichert. Diese Repositorien basieren typischerweise auf einer Datenbank oder XML-basierten Austauschformaten wie dem zuvor eingeführten PNML, bieten aber prozessspezifische Funktionalitäten (Yan u. a. 2012a). Zu diesen Funktionalitäten zählen beispielsweise spezielle Indices, die eine Klassifikation von Prozessen nach Fachbereichen in einem Unternehmen erlauben, Datenspeicherung zu Prozessausführungen oder eine ähnlichkeitsbasierte

Suche nach Prozessmodellen. Interessanterweise bestehen laut Elias und Johannesson (2012) weiterhin Herausforderungen bezüglich der Suchmöglichkeiten in solchen Bibliotheken, sodass die in den folgenden Kapiteln erläuterten Suchfunktionalitäten einen Beitrag zur Lösung dieser Problematik liefern können.

Abbildung 2.4 illustriert eine typische Schichtenarchitektur für Prozessmodell-Repositorien (Yan u. a. 2012a). Die Präsentationsschicht beinhaltet dabei die grafische Benutzeroberfläche, während die Managementschicht die prozessspezifischen Funktionalitäten wie etwa Import und Export von Modellen sowie die Suche nach Modellen umfasst. Die DBMS- und Storage-Schichten enthalten typische Datenbankfunktionalitäten wie Transaktionsverwaltung und die eigentliche Speicherung von Modellen und zugehörigen Daten. Zudem existieren üblicherweise Schnittstellen zu externen Tools, die z. B. die Modellierung und Analyse von Prozessmodellen erlauben.

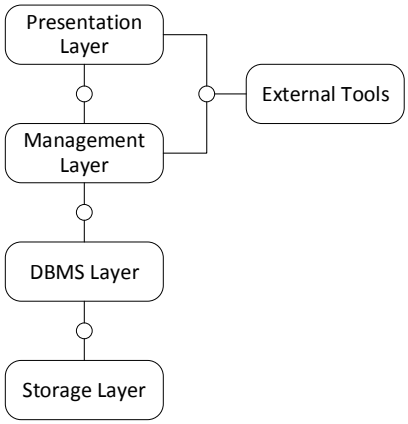


Abb. 2.4: Typische Architektur von Prozessmodell-Repositorien (in Anlehnung an (Yan u. a. 2012a))

In den folgenden Kapiteln werden somit verschiedene Suchfunktionalitäten beschrieben, die sich der Managementschicht zuordnen lassen. Dabei wird jedoch nicht auf einem kompletten Prozessmodell-Repositorium aufgebaut, sondern die Suchfunktionalitäten sind unabhängig davon, könnten jedoch in ein solches Repositorium integriert werden (eine Möglichkeit dazu wird in Kapitel 7 vorgestellt). Die Suche nach ähnlichen Aktivitäten (Kapitel 4) sowie die Suche nach ähnlichen Modellen (Kapitel 5) verwenden für die prototypische Implementierung PNML-Dateien als Grundlage. Für die Beschreibung der Algorithmen zur Berechnung von Anfrageergebnissen

mit der PNQML wird hingegen auf eine datenbankbasierte Prozessmodellbibliothek zurückgegriffen (Kapitel 6.3).

2.2 Geschäftsprozessmodelle und ihre Ähnlichkeit

Der Begriff *Ähnlichkeit* wird typischerweise im Kontext einer bestimmten Anwendung definiert. So bewertet etwa die Kosinus-Ähnlichkeit (van Rijsbergen 1979, S. 25) die Ähnlichkeit von Vektoren anhand der Größe des zugehörigen Winkels und das semantische Ähnlichkeitsmaß für Wörter von Wu und Palmer (1994) basiert auf der Distanz von Wörtern in einer Taxonomie. Lin (1998) hingegen definiert die Ähnlichkeit von zwei beliebigen Objekten A und B unter Zuhilfenahme dreier allgemeiner intuitiver Annahmen:

- **Annahme 1:** Je mehr Gemeinsamkeiten zwei Objekte A und B aufweisen, desto größer ist ihre Ähnlichkeit.
- **Annahme 2:** Je mehr Differenzen zwei Objekte A und B aufweisen, desto geringer ist ihre Ähnlichkeit.
- **Annahme 3:** Unabhängig von ihren Gemeinsamkeiten erhalten zwei Objekte A und B den höchsten Ähnlichkeitswert, wenn sie identisch sind.

Auch im Kontext der Berechnung der Ähnlichkeit von Prozessmodellen oder zwischen deren Aktivitäten wird im Folgenden auf diese intuitiven Annahmen zurückgegriffen. Dazu wird versucht anhand verschiedener Merkmale Gemeinsamkeiten und Unterschiede zu ermitteln, um einen Ähnlichkeitswert zu bestimmen. Im nächsten Kapitel 2.2.1 wird daher zunächst beschrieben, welche Wertebereiche für Ähnlichkeitsberechnungen typischerweise verwendet werden. Anschließend wird in Kapitel 2.2.2 auf die Merkmale zur Bestimmung von Gemeinsamkeiten und Unterschieden, den sogenannten Dimensionen, eingegangen. Schließlich erfolgt in Kapitel 2.2.3 eine Übersicht der Einsatzzwecke von Ähnlichkeitsberechnungen.

2.2.1 Quantifizierung von Ähnlichkeit

Ein Ähnlichkeitsmaß drückt generell die *Nähe* zwischen zwei Objekten bzw. in diesem Kontext von zwei Prozessmodellen oder deren Aktivitäten aus. Prinzipiell existieren verschiedene potentiell anwendbare Skalen zur Quantifizierung dieser Nähe. Dazu

zählen die Nominalskala, die Ordinalskala, die Intervallskala und die Verhältnisskala. Die Nominalskala erlaubt lediglich die Wahl zwischen der binären Unterscheidung, ob zwei Modelle ähnlich (Ähnlichkeitsgrad 1) oder unähnlich (Ähnlichkeitsgrad 0) sind. Diese Skala nutzt insofern nur die Extremwerte, wobei ein Ähnlichkeitsgrad von 1 auch als Äquivalenz aufgefasst werden kann (Wombacher und Rozie 2006). Im Kontext der Ähnlichkeit von Prozessmodellen wird diese Skala z. B. für Äquivalenzvergleiche von Beschriftungen eingesetzt. Die Ordinalskala hingegen beschreibt einen Grad an Ähnlichkeit basierend auf einer Charakteristik mit verschiedenen unterscheidbaren Ausprägungen, die in einer bestimmten Ordnungsbeziehung zueinander stehen. Diese Skala wird bislang kaum zur Berechnung der Ähnlichkeit von Prozessmodellen oder deren Elementen berücksichtigt. Lediglich Yan u. a. (2010) nutzen eine solche Skala zur Einordnung der Ähnlichkeit von Modellen. Mögliche Ausprägungen können etwa „keine Ähnlichkeit“, „geringe Ähnlichkeit“, „hohe Ähnlichkeit“ und „Gleichheit“ sein. Zudem wird eine solche Kategorisierung auch für das Prozessmodell-Matching kurz in (Fengel 2014) diskutiert.

Bei der Intervallskala kann der Differenzgrad zwischen Ausprägungen berücksichtigt werden. Diese Skala wird für Distanzberechnungen zwischen Prozessmodellen verwendet. Schließlich wird die Verhältnisskala dazu eingesetzt die Ähnlichkeit von Prozessmodellen innerhalb einer unteren und oberen Schranke zu quantifizieren. Typischerweise wird dazu das Intervall $[0, 1]$ verwendet, wobei 0 für minimale Ähnlichkeit bzw. Unähnlichkeit und 1 für maximale Ähnlichkeit bzw. Äquivalenz oder Gleichheit steht. Ein weiteres Beispiel für die Verwendung dieser Skala ist die Berechnung einer String-Editier Distanz zwischen Beschriftungen. In diesem Fall würde eine Distanz von 0 bedeuten, dass die Beschriftungen identisch sind, wohingegen der höchstmögliche Distanzwert von der Länge der Beschriftungen abhängt.

Somit wird üblicherweise die Verhältnisskala für die Quantifizierung der Ähnlichkeit von Prozessmodellen und ihren Aktivitäten in einem *metrischen Raum* verwendet. Ein metrischer Raum ist dabei folgendermaßen definiert (Zezula u. a. 2006):

Definition 2.8: Metrischer Raum, Metrik

Gegeben sei ein metrischer Raum $\mathcal{M} = (\mathcal{D}, d)$, der für eine Domäne von Objekten $\mathcal{D}$ und eine (Distanz-)Funktion d definiert ist. Dann sind die Eigenschaften dieser Funktion $d : \mathcal{D} \times \mathcal{D} \rightarrow \mathbb{R}$ charakterisiert als:

$\forall x, y \in \mathcal{D}, d(x, y) \geq 0$	Nichtnegativität,
$\forall x, y \in \mathcal{D}, d(x, y) = d(y, x)$	Symmetrie,
$\forall x, y \in \mathcal{D}, x = y \Leftrightarrow d(x, y) = 0$	Identität,
$\forall x, y, z \in \mathcal{D}, d(x, z) \leq d(x, y) + d(y, z)$	Dreiecksungleichheit

Die Nichtnegativitätseigenschaft sagt aus, dass ein Distanzwert für zwei Prozessmodelle bzw. zwei Modellelemente nicht kleiner als 0 sein darf, d. h. es sind nur positive Werte zulässig. Die Symmetrie-Eigenschaft kann für eine Beschleunigung verschiedener Kombinationen von Distanzberechnungen genutzt werden. Da die Reihenfolge in der zwei Modelle x und y miteinander verglichen werden bei einer symmetrischen Distanzfunktion keine Rolle spielt, muss die Distanz von x und y nur einmal berechnet werden, selbst wenn auch die Distanz von y und x relevant sein sollte. Die Identitätseigenschaft besagt weiterhin, dass zwei Modelle oder Modellelemente nur eine Distanz von 0 aufweisen sollen, wenn sie gleich sind. Die Eigenschaft Dreiecksungleichheit drückt schließlich aus, dass für drei Modelle x, y, z gelten soll, dass die Distanz von x und z höchstens so groß ist wie die addierten Distanzen zwischen x und y sowie y und z.

Repräsentiert $\mathbf{M}$ eine Prozessmodell-Bibliothek (die Domäne), wird die Intervallskala typischerweise für die Messung der Distanz $d : \mathbf{M} \times \mathbf{M} \rightarrow \mathbb{R}_0^+$ zwischen zwei Prozessmodellen aus $\mathbf{M}$ oder ihren Aktivitäten verwendet. Die Verhältnisskala wird hingegen zur Messung eines Ähnlichkeitsgrads $sim : \mathbf{M} \times \mathbf{M} \rightarrow [0, 1]$ eingesetzt. Ähnlichkeitswerte können zudem typischerweise in Distanzwerte transformiert werden.[3] Wie jedoch in (Kunze u. a. 2011a) beschrieben, erfüllen einige existierende Ähnlichkeitsmaße für Prozessmodelle die zuvor erwähnten Eigenschaften nicht vollständig. Von elf analysierten Maßen erfüllten nur drei alle Eigenschaften. Maße, die die Dreiecksungleichheit nicht erfüllen, stellen daher Semi-Metriken dar; Maße, die die Identitätseigenschaft verletzten ($d(x, y) = 0$ für $x \neq y$), stellen Pseudo-Metriken dar; und Maße, die die Symmetrie-Eigenschaft verletzen, stellen Quasi-Metriken dar.

Allerdings können je nach Einsatzzweck der Ähnlichkeitsberechnung Gründe dafür vorliegen, gewisse Eigenschaften zu verletzen. Dies kann beispielsweise bei der Su-

[3] Siehe etwa die Ausführungen in (Becker und Laue 2012) für die Transformation verschiedener Ähnlichkeitsmaße von Prozessmodellen.

che nach ähnlichen Prozessmodellteilen in einer Menge von Prozessmodellen der Fall sein, wobei auf das Enthaltensein eines Modellteils in einem Modell abgezielt wird. Dabei wird ein Prozessmodellteil als Anfrage verwendet und es sollte ein Ähnlichkeitswert von 1 berechnet werden, wenn ein Prozessmodell genau dieses Teil enthält, selbst wenn das Modell noch zusätzliche Elemente beinhaltet. Wird hingegen das vollständige Modell als Anfrage verwendet, sollte sich ein Ähnlichkeitswert kleiner als 1 ergeben, d. h., dass bei einem solchen Einsatzzweck die Symmetrie-Eigenschaft nicht erfüllt werden sollte.

2.2.2 Dimensionen der Ähnlichkeitsberechnung

Zur Berechnung der Ähnlichkeit von Prozessmodellen oder Aktivitäten lassen sich verschiedene Bestandteile eines Modells heranziehen (eine Übersicht ist in Abbildung 2.6 dargestellt). Auch wenn diese Dimensionen keine Dimensionen in einem strengen mathematischen Sinne sind, wird der Begriff *Dimension* verwendet, um zu verdeutlichen, dass jeweils ein anderer Bestandteil bzw. Aspekt eines Modells verwendet wird. Daher wird zur Quantifizierung der Ähnlichkeit von Modellen oder Modellelementen auch häufig eine Kombination dieser Dimensionen eingesetzt. Dies ist in Abbildung 2.5 mit drei Modellen illustriert, die zwei Mal einen Verkaufsprozess und ein Mal einen Kaufsprozess darstellen (siehe auch (Schoknecht u. a. 2017b)).

In *Verkäuferprozess 1* und *Verkäuferprozess 2* sind den Transitionen (Aktivitäten) identische Beschriftungen zugeordnet, sodass sich sinnvollerweise ein Ähnlichkeitswert von 1 ergibt, wenn nur die Dimension der natürlichen Sprache betrachtet wird. Allerdings sind sowohl ihre Graphstruktur als auch ihr Ablaufverhalten leicht unterschiedlich. So wird z. B. in *Verkäuferprozess 1* die Aktivität „Sende Rechnung“ immer vor der Aktivität „Liefere Produkte“ ausgeführt, wohingegen dies nicht notwendigerweise auf *Verkäuferprozess 2* zutrifft. Demzufolge könnte der letztendliche Ähnlichkeitswert dieser zwei Modelle kleiner als 1 sein.

Betrachtet man hingegen die Modelle *Verkäuferprozess 2* und *Käuferprozess*, so sind diese hinsichtlich der Graphstruktur identisch. Folglich sollte sich ebenso ein Ähnlichkeitswert von 1 ergeben, wenn nur diese Dimension berücksichtigt wird. Allerdings fallen Unterschiede auf, wenn die Dimension der natürlichen Sprache bei den Beschriftungen berücksichtigt wird. So unterscheiden sich z. B. die Beschriftungen „Erhalte Auftrag“ und „Erteile Auftrag“, was ebenso in einem letztendlichen Ähnlichkeitswert kleiner 1 resultieren sollte. Weitere Beispiele für Ähnlichkeitsabschätzungen

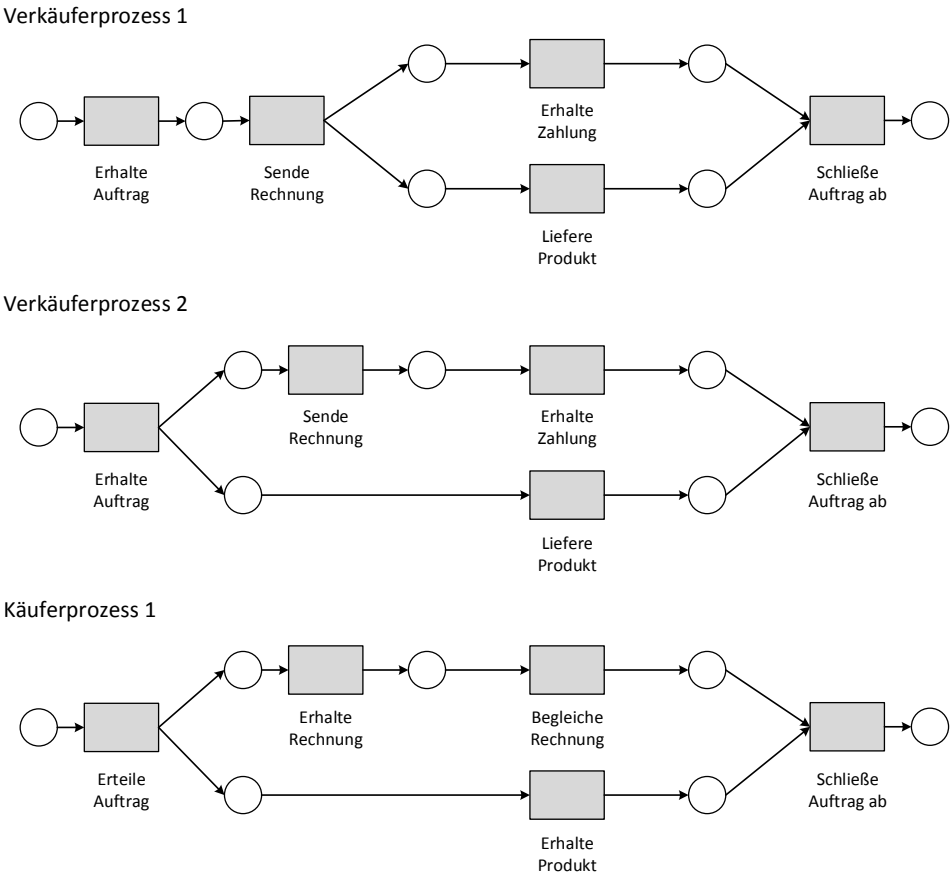


Abb. 2.5: Beispiel für die zur Ähnlichkeitsberechnung von Prozessmodellen oder Aktivitäten verwendeten Dimensionen (in Anlehnung an (Schoknecht u. a. 2017b))

unter Berücksichtigung der Graphstruktur- und Verhaltensdimension finden sich in (van der Aalst u. a. 2006).

Die in Abbildung 2.6 dargestellten Dimensionen werden im Folgenden zunächst erläutert und anschließend teilweise zur Berechnung von Ähnlichkeitswerten in den Suchfunktionalitäten eingesetzt. Die Abgrenzung der Dimensionen basiert auf der in (Schoknecht u. a. 2017b; Thaler u. a. 2017) grundsätzlich beschriebenen Einteilung.

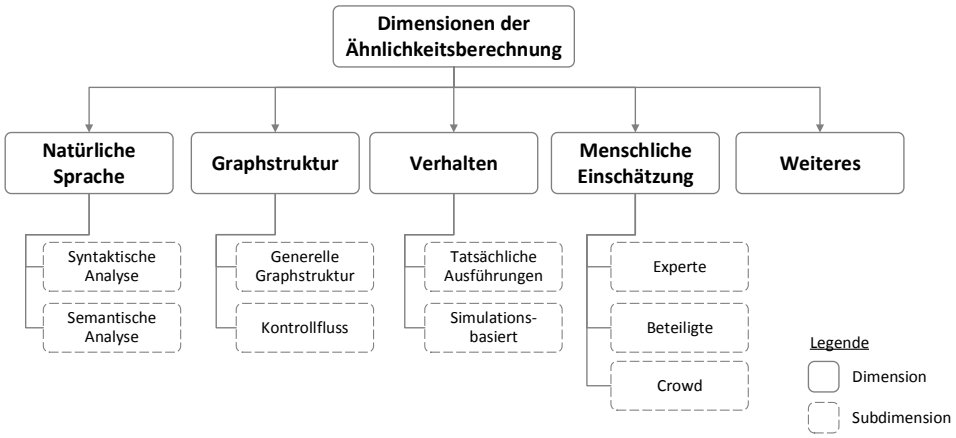


Abb. 2.6: Dimensionen der Ähnlichkeitsberechnung von Prozessmodellen (Schoknecht u. a. 2017b)

- **Natürliche Sprache:** Natürlichsprachliche Bestandteile wie der Modellname, die Beschriftungen von Modellelementen oder zusätzliche textuelle Beschreibungen von Prozessabläufen sind im Generellen von großer Bedeutung zum Verständnis von Prozessmodellen. So kann sich z. B. ein Mitarbeiter über die während eines Geschäftsprozesses anfallenden Aktivitäten anhand der Beschriftungen von Elementen eines entsprechenden Modells informieren. Auch verwandte Arbeiten zur Berechnung der Ähnlichkeit von Prozessmodellen nutzen insbesondere textuelle Beschriftungen in Modellen. Dabei werden diese sowohl aus syntaktischer als auch aus semantischer Sicht analysiert. Hinsichtlich der syntaktischen Analyse werden häufig String-Editier Distanzen genutzt, während die semantische Analyse darauf abzielt die Bedeutung einer Beschriftung zu verstehen, um letztlich die Ähnlichkeit von Modellen zu quantifizieren.[4]
- **Struktur:** In Bezug auf die Strukturdimension kann zum einen die generelle Graphstruktur eines Modells für die Berechnung eines Ähnlichkeitswerts herangezogen werden, ohne dabei die in Prozessmodellen vorkommenden Kontrollflusskonnektoren oder den Ablauf eines Prozesses gesondert zu berücksichtigen. Dabei wird von den spezifischen Aspekten eines Prozessmodells abs-

[4] Für detaillierte Ausführungen zu den eingesetzten Techniken dieser und der weiteren Dimensionen wird auf die Kapitel 4.2 und 5.2 verwiesen.

trahiert und lediglich die Repräsentation als Graph bestehend aus Knoten und Kanten für die Ähnlichkeitsberechnung eingesetzt. Techniken zur Berechnung eines Ähnlichkeitswerts aus dieser Perspektive sind z. B. die Berechnung einer Graph-Editier Distanz oder die Größe des größten gemeinsamen Teilgraphen. Zum anderen können die speziellen Kontrollflusskonnektoren in Prozessmodellen auch gesondert berücksichtigt werden, indem etwa die Position und Art dieser Konnektoren verglichen wird.

- **Verhalten:** Diese Dimension zur Messung der Ähnlichkeit fokussiert auf Prozessinstanzen und somit auf den Ausführungstraces bzw. dem Ausführungsverhalten eines Prozesses (siehe Definition 2.5). Diese Traces können von tatsächlich ausgeführten Prozessinstanzen stammen oder durch Simulationsläufe eines Prozessmodells generiert werden. Zur Berechnung eines Ähnlichkeitswerts kann z. B. die Anzahl an möglichen übereinstimmenden Traces herangezogen werden.

- **Menschliche Einschätzung:** Ein weiterer Aspekt ist die menschliche Einschätzung der Ähnlichkeit von Prozessmodellen und deren Elementen. Personen sind in der Lage die Ähnlichkeit subjektiv basierend auf ihrem individuellen Wissen zu quantifizieren. Hinsichtlich einer Teilautomatisierung könnte beispielsweise ein Maschinenlernverfahren Eingaben von Nutzern erhalten, ob ein automatisch bestimmter Ähnlichkeitswert korrekt ist oder nicht. Dabei werden sich die menschlichen Einschätzungen je nach Hintergrund einer Person unterscheiden. Prozessexperten werden aufgrund ihres Modellierungswissens und des Prozessverständnisses einen Ähnlichkeitsgrad festlegen, wohingegen Prozessbeteiligte auf Basis ihrer subjektiven Wahrnehmung des Prozesses entscheiden werden. Die Crowd als weitere Gruppe wird lediglich anhand ihres Verständnisses von Prozessmodellen und zugehörigen Beschreibungen einen Ähnlichkeitswert festlegen (siehe auch die Experimente in (Rodriguez u. a. 2016)).

- **Weiteres:** Zudem finden sich in der Literatur noch weitere Möglichkeiten, wie ein Ähnlichkeitswert bestimmt werden kann. Dies sind jedoch meist Einzelfälle, weshalb dafür keine eigene Kategorie erstellt wird. Beispiele dafür sind der *AML-PM* Ansatz (Antunes u. a. 2015) für das Prozessmodell-Matching, der ein Prozessmodell in eine ontologische Repräsentation überführt, sowie (Rinderle-Ma und Kabicher-Fuchs 2016), die anhand von übereinstimmenden Geschäftsregeln die Ähnlichkeit von Prozessmodellen bestimmen.

2.2.3 Ziele der Ähnlichkeitsberechnung

Neben der Wiederverwendung und Suche nach Prozessmodellen werden Ähnlichkeitsberechnungen noch für verschiedene weitere Zwecke eingesetzt. Abbildung 2.7 enthält dazu eine Übersicht, die im Folgenden näher erläutert wird.

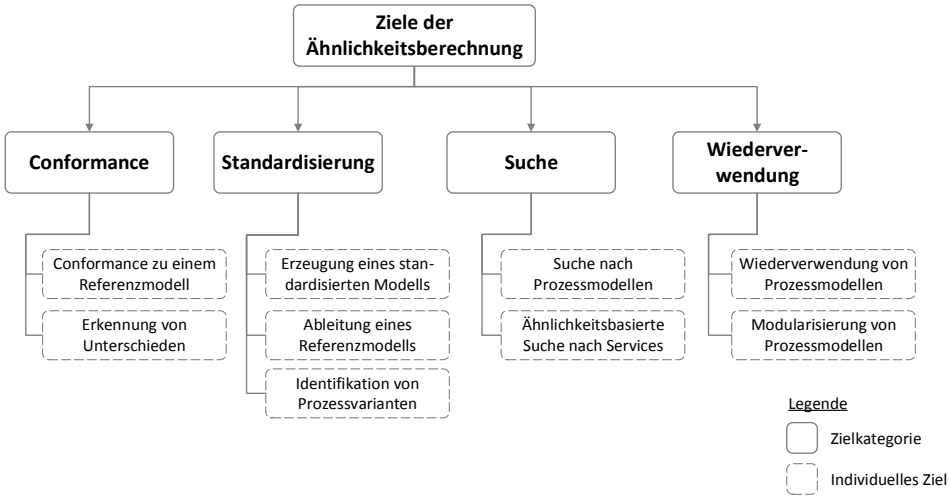


Abb. 2.7: Ziele von Ähnlichkeitsberechnungen (Schoknecht u. a. 2017b)

Conformance betrifft die Entsprechung eines Modells zu einem anderen Modell. Diese Kategorie umfasst beispielsweise die Ähnlichkeitsbestimmung zwischen einem Prozessmodell und einem zugehörigen Referenzmodell, sodass quantifiziert wird, inwieweit das Modell dem Referenzmodell entspricht. Ein weiteres Ziel dieser Kategorie ist die Erkennung von Abweichungen zwischen Prozessausführungen und einem zugehörigen Modell, sodass diese analysiert werden können.

Die zweite Kategorie *Standardisierung* verwendet Ähnlichkeitsberechnungen zur Vereinheitlichung von Prozessmodellen. Dies kann etwa im Kontext der Restrukturierung von Prozessvarianten zu einem Modell relevant sein, um automatisiert Modelle entdecken zu können, die standardisiert werden können. Ein weiterer Anwendungsfall dieser Kategorie ist die induktive Erzeugung von Referenzmodellen, bei der z. B. aus verschiedenen Modellen die besten Modellteile zu einem Referenzmodell zusammengefügt werden.

Die dritte Kategorie *Suche* wird schließlich in den Kapiteln 4 und 5 aufgegriffen, in denen Ansätze beschrieben werden, um ähnliche Aktivitäten in Prozessmodellen bzw. ähnliche Modelle in einer Modellbibliothek zu finden. Als weiteres Ziel dieser Kategorie kann auch die Suche nach Web Services gelten. Sollte eine Service-Beschreibung als Prozessmodell verfügbar sein, so können ähnlichkeitsbasierte Suchverfahren zum Auffinden von Web Services verwendet werden.

Schließlich wird die *Wiederverwendung* selbst häufig als Ziel der Entwicklung von Ähnlichkeitsmaßen genannt. Dies kann beispielsweise über eine Empfehlungsfunktion bei der Modellierung (Koschmider u. a. 2011) oder wie im Folgenden ausgeführt durch Suchfunktionalitäten realisiert werden. Ein weiterer Anwendungsfall in dieser Kategorie ist die Modularisierung von Modellen. Dabei sollen ähnliche Teilgraphen in verschiedenen Modellen entdeckt werden, sodass diese in ein separates Modell ausgelagert werden können, um etwa die Konsistenz und Wartbarkeit der Modelle sicherzustellen.

2.3 Zusammenfassung

In diesem Kapitel wurden zunächst Hintergründe zum Geschäftsprozessmanagement sowie Grundlagen zur Modellierung von Geschäftsprozessen beschrieben. Dabei wurde vertieft auf die Abbildung von Geschäftsprozessen durch Petri-Netze eingegangen. Als Grundlage für die in den Kapiteln 4 und 5 beschriebenen Suchfunktionen werden die in Definition 2.7 festgelegten *beschrifteten Workflow Netze* verwendet. Die Modelle werden typischerweise durch das XML-basierte Datenformat PNML (Weber und Kindler 2003) oder innerhalb von Prozessmodellbibliotheken (siehe Kapitel 2.1.3) gespeichert. Für die entwickelten Verfahren zum Prozessmodell-Matching und zur ähnlichkeitsbasierten Suche wird auf die PNML-Speicherung zurückgegriffen, während die Anfragesprache PNQML auf einer Speicherung von Modellen in einer Datenbank aufbaut.

Schließlich wurde auf den Begriff der *Ähnlichkeit* von Prozessmodellen eingegangen. Die Ähnlichkeit wird typischerweise durch einen Wert im Intervall $[0, 1]$ quantifiziert, indem verschiedene Aspekte eines Prozessmodells herangezogen werden. Die in Kapitel 2.2.2 als Dimensionen bezeichneten Aspekte umfassen die natürlichsprachlichen Beschriftungen von Modellelementen, die Graphstruktur und das Verhalten von Prozessmodellen sowie potentiell weitere spezifische Aspekte. Grundsätzlich versuchen

Ansätze zur Bestimmung eines Ähnlichkeitswerts Übereinstimmungen bzw. Differenzen hinsichtlich dieser Dimensionen zu identifizieren und zu quantifizieren. Neben der Suche nach Prozessmodellen zur Wiederverwendung werden Ähnlichkeitsberechnungen auch zu weiteren Zwecken wie etwa der Conformance-Überprüfung oder Standardisierung von Modellen durchgeführt.

Kapitel 3

Ähnlichkeitsberechnung von Prozessmodellen

In diesem Kapitel werden zunächst Konzepte und Techniken aus dem *Natural Language Processing* (NLP) Bereich bzw. der Linguistik vorgestellt, die im Folgenden in den einzelnen Suchfunktionalitäten verwendet werden, um Ähnlichkeitswerte zu berechnen. Die Verarbeitung natürlicher Sprache ist im Kontext von Suchfunktionen von Relevanz, um beispielsweise die Ähnlichkeit von Beschriftungen in Modellen zu berechnen. Eine Überprüfung auf exakte Übereinstimmung von Beschriftungen ist dabei nur begrenzt hilfreich, da typischerweise mehrere Personen Modelle erstellen und somit eine unterschiedliche Wortwahl zu erwarten ist. Zudem existieren mehrere Stile zur Beschriftung von Modellelementen (Mendling u. a. 2010), in denen Verben etwa substantiviert sein können oder nicht. Schließlich werden noch die Latent Semantic Analysis aus dem Information Retrieval sowie verschiedene Maßzahlen zur Bewertung der Qualität von Suchergebnissen eingeführt.

Dieses Kapitel beginnt in Abschnitt 3.1 mit der Vorstellung von grundlegenden Konzepten und Techniken aus dem NLP-Bereich. Anschließend wird in Kapitel 3.2 auf die Ähnlichkeit von Wörtern eingegangen. Dabei wird insbesondere ein Schwerpunkt auf die Quantifizierung der Ähnlichkeit bzw. Verwandtschaft einzelner Wörter gelegt. Im dritten Teilkapitel wird die Latent Semantic Analysis vorgestellt, die die Grundlage für die in Kapitel 5 beschriebene ähnlichkeitsbasierte Suche nach Prozessmodellen darstellt. Schließlich werden in Abschnitt 3.4 Maßzahlen zur Bewertung von Suchergebnissen erläutert, die für die Evaluation der Suchverfahren eingesetzt werden.

3.1 Techniken zur Verarbeitung natürlicher Sprache

Das Forschungsfeld *Natural Language Processing* (NLP) befasst sich mit der Verarbeitung von Text und Sprache durch Computer, um beispielsweise Interaktionen zwischen Menschen und Maschinen oder die automatisierte Übersetzung von Texten zu ermöglichen (Jurafsky und Martin 2009). Die folgenden Abschnitte führen kurz in Techniken aus verschiedenen NLP-Bereichen ein, die zur Berechnung von Ähnlichkeitswerten verwendet werden. Für weiterführende Details wird auf die in den Abschnitten angegebene Literatur sowie auf (Jurafsky und Martin 2009) verwiesen.

3.1.1 Zerlegung von Wortfolgen

Eine Technik zur Verarbeitung natürlicher Sprache, die in den folgenden Kapiteln zur Ähnlichkeitsberechnung eingesetzt wird, betrifft die Zerlegung von Wortfolgen – die sogenannte *Tokenisierung*. Diese wird benötigt, um Beschriftungen von Prozessmodellelementen in ihre einzelnen Wörter zerlegen zu können. So kann beispielsweise die Beschriftung „Deliver the package“[1] in die einzelnen Worte „Deliver“, „the“ und „package“ zerlegt werden, um die Ähnlichkeit einzelner Wörter berechnen zu können (siehe hierzu auch Kapitel 3.2). Dabei kann ein Wort als eine alphanumerische Einheit aufgefasst werden, die sowohl links als auch rechts durch Leerzeichen oder Interpunktion begrenzt ist (Carstensen u. a. 2010, S. 264).

Die Zerlegung erscheint zunächst klar abgegrenzt und relativ trivial, allerdings existieren Fälle, in denen eine Zerlegung nicht erwünscht ist bzw. es nicht eindeutig ist, was eine passende Zerlegung ist (Manning u. a. 2008, S. 23ff.). Punkte im Zusammenhang mit Abkürzungen wie „e. g.“ stellen etwa tendenziell unerwünschte Zerlegungspunkte dar. Im Falle von Apostrophen muss entschieden werden, was eine sinnvolle Tokenisierung für einen konkreten Anwendungsfall ist (z. B. Zerlegung von „O’Neill“ in „O“ und „Neill“ oder in „O’“ und „Neill“ (Manning u. a. 2008, S. 23)).

Die Zerlegung von Beschriftungen in einzelne Wörter ist im Kontext der Ähnlichkeitsberechnung sinnvoll (siehe dazu auch die Ausführungen in Kapitel 3.2), sodass ein Verfahren zur Tokensisierung eingesetzt werden muss. Für die Implementierungen

[1] Die Ausführungen und Beispiele in diesem und den folgenden Abschnitten beziehen sich vorrangig auf die englische Sprache, da die zur Evaluation der Suchverfahren verwendeten Modelle englische Beschriftungen enthalten. Für Deutsch existieren ebenso entsprechende Konzepte und Implementierungen.

der Suchverfahren wurde dazu der Stanford Parser verwendet (Klein und Manning 2003), dessen Tokenizer Komponente[2] verschiedene Optionen bietet sowie für verschiedene Sprachen einsetzbar ist. Die Qualität der Suchverfahren hängt somit aber auch von dem Einsatz und der Konfiguration von Techniken und Software zur Verarbeitung natürlicher Sprache ab.

3.1.2 Stoppwörter

Als *Stoppwörter* werden Worte bezeichnet, die keine oder eine geringe Bedeutung für die Erfassung eines Dokumenteninhalts besitzen oder die für die Beantwortung eines Informationsbedürfnisses wenig hilfreich sind (Manning u. a. 2008, S. 27). Diese wurden ursprünglich von Luhn (1960) zur Indexierung von Bibliographiedaten eingeführt, um irrelevante Begriffe von der Indexierung auszuschließen. Ein typisches Merkmal solcher Stoppworte ist, dass sie sehr häufig vorkommen und wenig zur Unterscheidung von Dokumenten beitragen. Beispiele für englische Stoppwörter sind in Tabelle 3.1 abgebildet.

Tab. 3.1: Auszug aus der englischen Stoppwort-Liste in Reuters-RCV1 (Lewis u. a. 2004)

a	able	about	again
an	and	are	asking
be	because	both	...

Solche Stoppwort-Listen können basierend auf Dokumentensammlungen ermittelt werden, indem beispielsweise die häufigsten Begriffe als Stoppwörter klassifiziert werden (Manning u. a. 2008, S. 27). Typischerweise müssen Stoppwort-Listen zudem an den jeweiligen Anwendungsbereich angepasst werden. So bestehen z. B. manche Musiktitel nur aus Stoppwörtern (*Let It Be*), sodass bei einer Indexierung von Musiktiteln darauf geachtet werden muss (Manning u. a. 2008, S. 27). Stoppwort-Listen finden sich heutzutage vielfach im Internet für verschiedene Sprachen,[3] sodass diese wiederverwendet werden können.

Für die in den folgenden Kapiteln erläuterten Suchverfahren werden Stoppwort-Listen verwendet, um für die Ähnlichkeitsberechnung von Beschriftungen nicht rele-

[2] http://nlp.stanford.edu/software/tokenizer.shtml.
[3] http://www.ranks.nl/stopwords.

vante Worte zu entfernen. Nicht relevant bedeutet in diesem Kontext vor allem, dass nach wie vor die drei typischen Bestandteile von Aktivitätsbeschriftungen – Aktion, Geschäftsobjekt und optionales Fragment (Leopold u. a. 2012) – erkenntlich bleiben. Wenn z. B. die beiden Beschriftungen „Deliver package“ und „Deliver the package“ betrachtet werden, so unterscheiden sie sich nur durch das (Stopp-)Wort „the“. Durch die Entfernung des Stoppworts in diesem Beispiel würde die Ähnlichkeitsbestimmung stark vereinfacht, da die Beschriftungen identisch wären. Dabei wird allerdings keine existierende Liste vollständig verwendet, da diese auch Verben enthalten, die jedoch typischerweise im Aktionsteil einer Beschriftung vorkommen. Daher sind Verben zum Vergleich und zur Unterscheidung von Beschriftungen der Aktivitäten in Prozessmodellen relevant und werden somit aus den Stoppwort-Listen entfernt.[4]

Neben Verben könnten potentiell auch andere Worte wie etwa „not“ oder „nothing“ aus den Stoppwort-Listen entfernt werden – z. B. um explizit den Gegensatz in den Beschriftungen „Deliver package“ und „Do not deliver package“ zu erkennen –, dies ergab aber bei den verwendeten Modelldatensätzen keinen Unterschied. Hieran lässt sich allerdings erkennen, dass je nach Anwendungsfall bzw. Modelldatensatz andere Stoppwort-Listen sinnvoll sein können.

3.1.3 Stammformreduktion von Wörtern

Die *Stammformreduktion* oder englisch das *Stemming* von Wörtern bezeichnet im Information Retrieval oder der Computerlinguistik häufig eingesetzte Verfahren, um unterschiedliche morphologische Varianten eines Worts auf ihren Wortstamm zurückzuführen. Ein konkretes Beispiel ist etwa die Zurückführung der englischen Worte „car“, „cars“ und „cars'“ auf ihren Stamm „car“. Ein weiteres Beispiel ist die Generierung von Verbstämmen aus konjugierten Verben. Die grundsätzliche Idee, Verfahren aus diesem Bereich für das Information Retrieval oder die Suche nach ähnlichen Prozessmodellen zu verwenden, besteht darin, unabhängig von der konkreten Form eines Suchbegriffs auch Dokumente bzw. Modelle zu finden, die Worte mit dem gleichen Wortstamm enthalten (Manning u. a. 2008, S. 32).

[4] Die für die Implementierungen verwendeten Listen finden sich online bei den jeweiligen Implementierungen. Es werden jeweils nur englische Stoppwort-Listen verwendet, da die in den Evaluationen verwendeten Modelle nur englische Beschriftungen aufweisen.

Generell existieren verschiedene Ansätze, die sich in die Klassen abschneidende bzw. regelbasierte Verfahren, statistische Verfahren und gemischte Verfahren einteilen lassen (Jivani 2011). Regelbasierte Verfahren wie der *Porter Stemmer* (Porter 1980) verwenden Regeln, um Wörter auf ihren Stamm zurückzuführen (z. B. die Reduktion von englischen Pluralen endend auf „IES“ nach „I“, sodass aus „ponies“ das Wort „poni“ erzeugt wird (Manning u. a. 2008, S. 33)). Verfahren aus dem statistischen Bereich verwenden etwa die Häufigkeit von Buchstabenfolgen bestimmter Länge (N-Gramme), um auf Wortstämme zu schließen und schließlich kombinieren gemischte Verfahren Ansätze aus den beiden anderen Kategorien.

In den folgenden Suchverfahren wird der Porter Stemmer verwendet, um Wörter auf ihren Stamm zurückzuführen, da er in einem Vergleich verschiedener Stemming-Verfahren sehr gut abgeschnitten hat (Jivani 2011) und auch im Information Retrieval Bereich generell als sehr effektiv gilt (Manning u. a. 2008, S. 33). Allgemein gilt allerdings auch für das Stemming, dass bei anderen Sprachen andere Verfahren besser geeignet sind bzw. benötigt werden, da etwa für die deutsche Sprache andere Regeln zur Umformung gelten.

3.2 Ähnlichkeit von Wörtern

In diesem Abschnitt werden verschiedene Ansätze vorgestellt, um die Ähnlichkeit einzelner Wörter berechnen zu können. Verfahren aus diesem Bereich werden insbesondere in Kapitel 4 verwendet, um die Ähnlichkeit von Beschriftungen in Prozessmodellen zu ermitteln. Generell lassen sich drei unterschiedliche Kategorien von Verfahren zur Berechnung der Ähnlichkeit von Wörtern unterscheiden. Zum einen existieren Verfahren, die Wörter als Zeichenketten auffassen und einen Ähnlichkeitswert anhand der Anzahl unterschiedlicher Zeichen ermitteln. Zum anderen wird versucht die Ähnlichkeit von Wörtern auf ihrer Bedeutungsebene (Semantik) zu quantifizieren (z. B. anhand von Synonymen). Zudem existieren Ansätze wie WORD2VEC (Mikolov u. a. 2013b), mit deren Hilfe Wortähnlichkeiten durch die Analyse einer großen Anzahl an Textdokumenten bestimmt werden können.

3.2.1 Ähnlichkeit von Zeichenketten

Die Ähnlichkeit von zwei Wörtern w_1 und w_2 kann basierend auf der minimale Anzahl an Editieroperationen zwischen diesen beiden Wörtern, die w_1 in w_2 transfor-

miert, bestimmt werden. Zu diesen Editieroperationen zählen das Einfügen und Löschen sowie die Ersetzung und Vertauschung einzelner Zeichen. Das wahrscheinlich bekannteste Verfahren in der Kategorie von Editier-Distanz Algorithmen wurde von Levenshtein (1966) beschrieben. Bei einer Berechnung zwischen den Wörtern „deliver“ und „delay“ mit dem in (Levenshtein 1966) publizierten Algorithmus beträgt die Editier-Distanz etwa 4. In „deliver“ werden dazu die Buchstaben i und v durch die Buchstaben a und y ersetzt sowie die beiden restlichen Buchstaben e und r gelöscht, sodass sich der Wert von 4 ergibt.

Aus einem solchen Distanzwert kann schließlich ein Ähnlichkeitswert im Intervall $[0, 1]$ ermittelt werden, indem die folgende Berechnung durchgeführt wird. Dabei bezeichnen $dist(w_1, w_2)$ die Distanz zwischen zwei Wörtern und $len(w_i)$ die Anzahl an Zeichen eines Worts:

$$sim(w_1, w_2) = \frac{dist(w_1, w_2)}{max(len(w_1), len(w_2))}.$$

Diese Art der Ähnlichkeitsbestimmung von Wörtern ist beispielsweise beim Auffinden ähnlicher Aktivitäten in Prozessmodellen anhand ihrer Beschriftungen hilfreich, indem kleinere Rechtschreib- oder Tippfehler erkannt werden können. So beträgt etwa die Distanz zwischen zwei Aktivitäten mit den Beschriftungen „deliver“ und „delivre“ bei Berücksichtigung von Zeichenvertauschungen 1. Damit ergibt sich immer noch ein Ähnlichkeitswert von 0,86. Wird etwa ein Schwellenwert von 0,8 festgelegt, ab dem Aktivitäten als übereinstimmend zu klassifizieren sind, so hätte der Tippfehler im Beispiel keine Auswirkung. Auch Wortänderungen wie bei der Konjugation von Verben oder durch Pluralbildung von Substantiven können somit zumindest teilweise erkannt werden. Ein Beispiel hierfür ist etwa die unterschiedliche Person der Verben „deliver“ und „delivers“.

Bei Beschriftungen mit mehreren Wörtern kann die Ähnlichkeitsberechnung dahingehend erweitert werden, dass zwischen jeder Kombination von Wörtern ein Ähnlichkeitswert berechnet wird. Anschließend werden die Kombinationen von Wörtern ausgewählt, die den Ähnlichkeitswert der vollständigen Beschriftung maximieren. Für die Beschriftungen „deliver package“ und „package delivery“ würden dazu die Ähnlichkeitswerte für die Kombinationen (deliver,delivery) = 0,86, (deliver,package) = 0,00, (package,package) = 1,00 und (package,delivery) = 0,00 berechnet. Somit ergibt sich als Summe der Wortähnlichkeitswerte der beiden Beschriftungen 1,86 bzw.

0,93 als durchschnittlicher Ähnlichkeitswert pro Wort. Daher kann trotz unterschiedlicher Wortreihenfolge und Verwendung des Substantivs „delivery“ anstatt des Verbs „deliver“ eine Übereinstimmung der Aktivitäten ermittelt werden.

3.2.2 Semantische Ähnlichkeit in Wortkorpora

Die semantische Ähnlichkeit von Wörtern kann als Ähnlichkeit von Konzepten in einer Taxonomie, Ontologie oder lexikalischen Datenbank wie WORDNET (Fellbaum 1998) aufgefasst werden (Resnik 1995). In solchen Datenbanken sind letztlich Wörter und ihre Beziehungen zueinander gespeichert. Dies wird im Folgenden beispielhaft für WORDNET illustriert, da diese Datenbank auch in einem der in Kapitel 4 beschriebenen Ansätze zum Prozessmodell-Matching eingesetzt wird.

In WORDNET werden Substantive, Verben, Adjektive und Adverbien der englischen Sprache als Konzepte aufgefasst, die über verschiedene semantische und lexikalische Beziehungen miteinander verbunden sind (Fellbaum 1998). Beispiele solcher Wortbeziehungen sind Synonymie, Hyponymie oder Meronymie (Miller 1995), die in verschiedenen Ansätzen dazu verwendet werden, die Ähnlichkeit von Wörtern zu quantifizieren. In der Version 3.0 von WORDNET sind 155.287 englische Wörter und ihre Beziehungen zueinander enthalten. Ein Auszug aus der WORDNET Hierarchie in Anlehnung an (Lin 1998) ist in Abbildung 3.1 dargestellt.

Exemplarisch werden im Folgenden drei Ansätze vorgestellt, die auf Basis einer lexikalischen Datenbank wie WORDNET die Ähnlichkeit von Wörtern quantifizieren. Resnik (1995) misst die Ähnlichkeit von Wörtern anhand ihres Informationsgehalts. Er geht dabei davon aus, dass Wörter in einer hierarchischen Is-a Beziehung vorliegen (z. B. „credit“ Is-a „medium of exchange“), die ein Wurzelkonzept enthält. Jedem Konzept bzw. Wort wird in dieser Hierarchie ein Wert für den Informationsgehalt zugewiesen. Die Ähnlichkeit zweier Wörter ergibt sich dann schließlich aus dem höchsten Informationsgehalt eines Konzepts, dass beide Wörter als Unterkonzepte enthält. Lin (1998) erweiterte diesen Ansatz, sodass nicht nur der Informationsgehalt des gemeinsamen Superkonzepts zweier Wörter berücksichtigt wird, sondern auch der Informationsgehalt dieser beiden Wörter selbst. Wu und Palmer (1994) hingegen messen die Ähnlichkeit von zwei Wörtern anhand der jeweiligen Knotenanzahl zu dem in einer Hierarchie niedrigsten gemeinsamen Konzept. Diese Ähnlichkeitsberechnung basiert somit nicht auf dem Informationsgehalt, sondern auf der Distanz von Wörtern zu einem gemeinsamen Superkonzept.

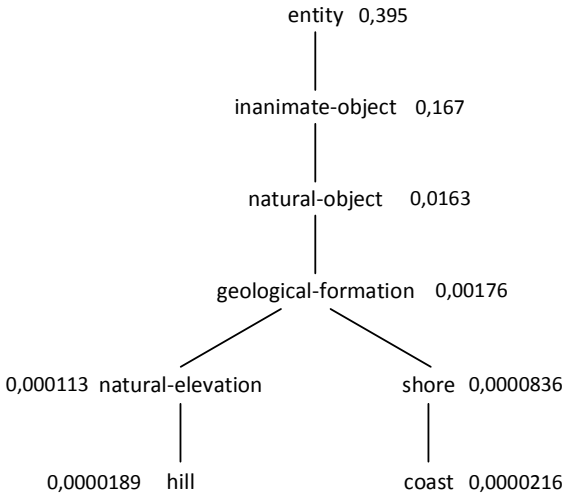


Abb. 3.1: Auszug aus der WORDNET Hierarchie nach (Lin 1998). Die zu den Konzepten annotierten Zahlen entsprechen den Wahrscheinlichkeiten dieser Konzepte.

Soll beispielsweise die Ähnlichkeit der Wörter „hill“ und „coast“ aus Abbildung 3.1 berechnet werden, so ergibt sich nach (Lin 1998) folgender Wert:

$$sim(\text{hill}, \text{coast}) = \frac{2 \times logP(\text{geological-formation})}{logP(\text{hill}) + logP(\text{coast})} = 0,59.$$

P gibt dabei die Wahrscheinlichkeit eines Terms an und die log-Funktion berechnet den Logarithmus. Das auf der gleichen Grundidee basierende Verfahren von Resnik (1995) resultiert für die Wörter „hill“ und „coast“ in einem Ähnlichkeitswert von $sim(\text{hill, coast}) = -logP(\text{geological-formation}) = 6,34$. Die Berechnung von Wu und Palmer (1994) beruht nicht auf dem Informationsgehalt, sondern auf der Distanz von Knoten in einer lexikalischen Datenbank. Dazu verwenden sie folgende Formel:

$$sim(C_1, C_2) = \frac{2 \times N_3}{N_1 + N_2 + 2 \times N_3}.$$

Dabei bezeichnen C_1 und C_2 zwei Terme aus einer Hierarchie, N_1 und N_2 bezeichnen die Anzahl an Kanten von C_1 und C_2 zu dem spezifischsten gemeinsamen Term C_g

und N_3 bezeichnet die Anzahl an Kanten von C_g zum Wurzelelement der Hierarchie. Im Beispiel ergibt sich somit:

$$sim(\text{hill}, \text{coast}) = \frac{2 \times 3}{2 + 2 + 2 \times 3} = 0,6.$$

Im Rahmen der Ähnlichkeit von Prozessmodellen ist die semantische Ähnlichkeit von Wörtern beispielsweise zur Erkennung von Synonymen in Beschriftungen von Relevanz. Zwei Beschriftungen „Deliver a speech“ und „Present a speech“ beschreiben etwa die gleiche Aktivität – das Vortragen einer Rede –, verwenden allerdings unterschiedliche Worte. In einem solchen Fall ist die Berechnung eines Ähnlichkeitswerts auf Zeichenebene nur von geringem Nutzen, da diese keine Beziehungen zwischen Worten berücksichtigt.

3.2.3 Word2Vec

Neben der Berechnung eines Ähnlichkeitswerts von Wörtern durch eine lexikalische Distanz oder mit Hilfe von Wortkorpora können dazu auch Vektorrepräsentationen verwendet werden (Mikolov u. a. 2013a). Diese Ansätze sind dem Bereich der statistischen Sprachenmodellierung zuzuordnen und lernen aus Trainingsdaten Wahrscheinlichkeitsfunktionen für das Auftreten eines Wortes bei gegebenen, vorausgegangen Wörtern (Bengio u. a. 2003). Dazu wird jedes Wort durch einen *Feature Vektor* dargestellt, der verschiedene Aspekte eines Worts repräsentiert und mit Hilfe von Trainingsdaten gelernt wird (Bengio u. a. 2003).

Zur Erzeugung von Feature Vektoren können u. a. auch die im nächsten Abschnitt vorgestellte Latent Semantic Analysis (Deerwester u. a. 1990) oder die Latent Dirichlet Allocation (Blei u. a. 2003) verwendet werden, der WORD2VEC Ansatz nutzt hingegen neuronale Netze, da diese in Experimenten eine höhere Qualität aufwiesen (Bengio u. a. 2003; Mikolov u. a. 2013a). In Abbildung 3.2 sind die in (Mikolov u. a. 2013a) beschriebenen Varianten neuronaler Netze zur Vorhersage von Wörtern dargestellt.

Bei der CBOW Variante wird das neuronale Netz darauf trainiert ein Wort $w(t)$ anhand der Wörter im Kontext vorherzusagen. Dazu werden sowohl Wörter verwendet, die vor $w(t)$ stehen, als auch Wörter, die nach $w(t)$ vorkommen. Die Skip-gram Variante kehrt dieses Vorgehen um, indem zu einem Wort $w(t)$ die Wörter im Kontext vorhergesagt werden. Die Ähnlichkeit von Wörtern kann mit den durch die neuronalen

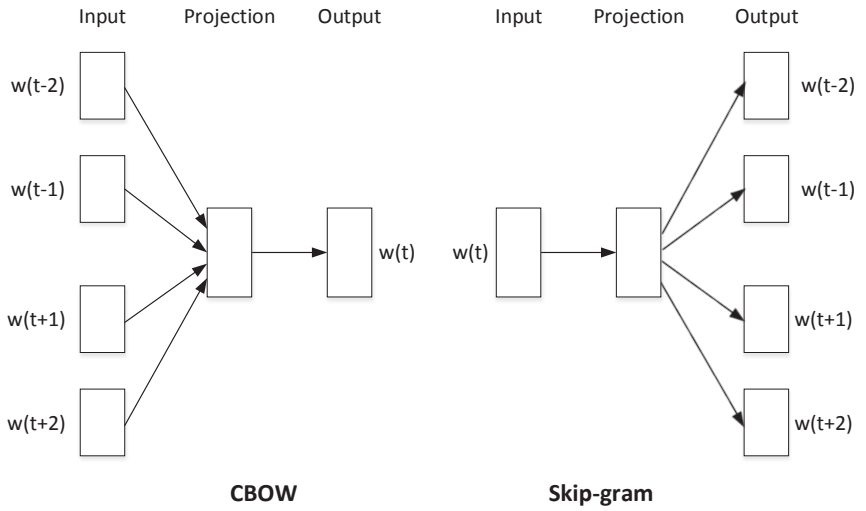


Abb. 3.2: Schematische Darstellung der neuronalen Netze zur Vorhersage von Wörtern aus (Mikolov u. a. 2013a)

Netze gelernten Vektoren daraufhin z. B. durch die Kosinus-Ähnlichkeit (van Rijsbergen 1979, S. 25) bestimmt werden. Für das Prozessmodell-Matching verwendet einer der im nächsten Kapitel vorgestellten Ansätze die Skip-gram Variante (Mikolov u. a. 2013b), da diese in Experimenten besser abgeschnitten hat (Mikolov u. a. 2013a).

3.3 Latent Semantic Analysis

Die *Latent Semantic Analysis* (LSA), teilweise auch als Latent Semantic Indexing bezeichnet, ist ein Verfahren aus dem Information Retrieval, um zu einem Anfragedokument ähnliche Dokumente in einer Dokumentensammlung aufzufinden (Deerwester u. a. 1990). Sie basiert auf einem mathematisch-statistischen Ansatz bei dem Wortbedeutungen bzw. die Bedeutung eines Textabschnitts anhand von Wortauswahl und Wortkombinationen beschrieben und erfasst werden (Landauer 2007). Die LSA geht über klassische Ansätze aus dem Information Retrieval hinaus, die auf einem syntaktischen Vergleich zwischen den Wörtern der Suchanfrage und Dokumenten beruhen. Die LSA ergänzt diesen rein syntaktischen, textbasierten Vergleich um einen semantischen Aspekt, der es ermöglicht, Bedeutungen der Suchwörter und eine „latente“

semantische Struktur der Wortnutzung bei der Suche nach relevanten Dokumenten miteinzubeziehen und damit Suchergebnisse zu verbessern (Deerwester u. a. 1990).

Die LSA gründet u. a. auf der Annahme, dass sich die Bedeutung eines Wortes aus den Bedeutungen der Textpassagen ergibt, in denen es vorkommt (Landauer 2007). Wörter ähneln sich somit in ihrer Bedeutung, wenn sie in ähnlichen Kontexten verwendet werden (sog. distributional hypothesis (Turney und Pantel 2010)). Die in Kapitel 3.2 beschriebenen Ansätze versuchen hingegen die Ähnlichkeit von Wörtern basierend auf einer lexikalischen Datenbank zu quantifizieren, was bei der LSA nicht notwendig ist. Die Bedeutung einer Textpassage wird wiederum (bis zu einem gewissen Grad) durch die Wörter, die sie enthält beschrieben (sog. bag-of-words hypothesis (Turney und Pantel 2010)). Mit Hilfe der in der LSA verwendeten mathematisch-statistischen Vorgehensweise werden semantische Strukturen und Beziehungen zwischen Wörtern aufgedeckt, sodass sich die Bedeutung eines Dokuments letztendlich aus den Einzelbedeutungen der darin enthaltenen Wörter ergibt. Landauer (2007) beschreibt deshalb die LSA als einen *Bag-of-Words* Ansatz im erweiterten Sinne: Die LSA basiert nicht auf einem mit dem Bag-of-Words Ansatz oft assoziierten herkömmlichen Vektormodell und damit einem ausschließlich syntaktischen Vergleich von Termen, sondern es wird auch die Bedeutung bzw. Semantik der Terme miteinbezogen.

Demzufolge zielt die LSA allgemein auf eine Analyse der Wortwahl ab, durch die eine latente semantische Struktur eines Textdokuments abgebildet ist. Zudem ergibt sich die durch die LSA formulierte Bedeutung aus einer Analyse der Beziehungen der Wörter und Textpassagen innerhalb des gesamten untersuchten Korpus (z. B. einer Dokumentenmenge) und nicht aus einer Referenzierung zu Realweltobjekten (Kintsch u. a. 2007). Es werden deshalb keine Thesauri, Ontologien oder ähnliches benötigt, um die Ähnlichkeit von Dokumenten zu bestimmen.

Das in der LSA verwendete Berechnungsverfahren basiert auf dem Konzept des Vektorraummodells (Salton u. a. 1975). Dieser Vektorraum wird durch eine sogenannte *Term-Dokument Matrix* repräsentiert. Mit dem Begriff Dokument werden dabei Textpassagen festgelegter Länge beschrieben, während der Begriff Term z. B. für einzelne Wörter aber auch für Sinneinheiten (etwa der Stadtname „Hong Kong“) steht. Abbildung 3.3 zeigt eine schematische Darstellung einer solchen Matrix. Die Spalten repräsentieren dabei die Dokumente $D_1, D_2, \cdots, D_n$, während die Zeilen die Terme $T_1, T_2, \cdots, T_t$ abbilden. Ein Matrixeintrag spiegelt das Gewicht $(w_{11}, \cdots, w_{tn})$ eines

bestimmten Terms in einem bestimmten Dokument wider. Dabei kann es sich z. B. um die Termhäufigkeit handeln.

$$\begin{array}{c} \\ T_1 \\ T_2 \\ \vdots \\ T_t \end{array} \begin{array}{c} \begin{array}{cccc} D_1 & D_2 & \cdots & D_n \end{array} \\ \begin{pmatrix} w_{11} & w_{12} & \cdots & w_{1n} \\ w_{21} & w_{22} & \cdots & w_{2n} \\ \vdots & \vdots & \cdots & \vdots \\ w_{t1} & w_{t2} & \cdots & w_{tn} \end{pmatrix} \end{array}$$

Abb. 3.3: Schematische Darstellung einer Term-Dokument Matrix

Die LSA generiert ihr Wissen folglich durch eine (statistische) Analyse großer Textmengen (Dumais 2007). Mit Hilfe von Singulärwertzerlegung[5] und Dimensionsreduktion wird aus der Term-Dokument Matrix ein sogenannter *semantischer Raum* erzeugt. Die darin enthaltenen, abgeleiteten Dimensionen können als künstliche Konzepte angesehen werden. Sowohl Terme als auch Dokumente werden in diesem semantischen Raum durch Gewichtsvektoren charakterisiert und repräsentiert, durch die letztlich die *Bedeutung* von Termen und Dokumenten ausgedrückt wird, wobei die Gewichte die Stärke der Assoziation zu den künstlichen Konzepten ausdrücken (Deerwester u. a. 1990). Die Ähnlichkeit von Termen oder Dokumenten kann dementsprechend z. B. mit Hilfe der Kosinus-Ähnlichkeit (van Rijsbergen 1979, S. 25) oder der Euklidischen Distanz der korrespondierenden Vektoren bestimmt werden.

3.3.1 Berechnungsverfahren der LSA

Abbildung 3.4 bildet die grundlegenden Schritte für das generelle Berechnungsverfahren der Latent Semantic Analysis auf Grundlage der Ausführungen in (Dumais 2007; Hu u. a. 2007) ab. Grundsätzlich muss zunächst eine Term-Dokument Matrix analog zu Abbildung 3.3 und anderen Vektorraumverfahren aufgebaut werden. Danach erfolgt der eigentliche Kern der LSA: Die Singulärwertzerlegung in Verbindung mit einer Abbildung von Dokumenten und Termen in einen Vektorraum geringerer

[5] Für eine Übersicht der mathematischen Entwicklung der Singulärwertzerlegung siehe etwa (Stewart 1993).

Dimensionalität (Dimensionsreduktion) (Dumais 2007). Im dadurch erzeugten sogenannten *semantischen Raum* können Ähnlichkeitsberechnungen zwischen Dokumenten bzw. Termen vorgenommen werden. Im Folgenden werden zunächst die mathematischen Grundlagen der LSA erläutert, bevor anschließend die einzelnen Schritte detaillierter ausgeführt werden.

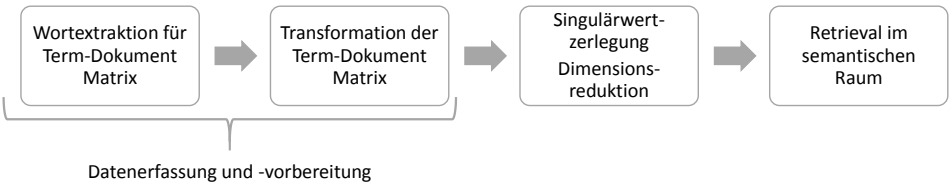


Abb. 3.4: Berechnungsschritte der Latent Semantic Analysis

Die LSA basiert auf einem Vektorraummodell, das durch eine Term-Dokument Matrix gemäß Abbildung 3.3 darstellbar ist und durch Verwendung linearer Algebra effektiv und automatisiert für das Information Retrieval genutzt werden kann (Martin und Berry 2007). Die Singulärwertzerlegung und die Dimensionsreduktion der LSA verkleinern diesen Vektorraum, um die latente semantische Struktur eines Dokuments aufzudecken, die in der Wortauswahl und -verwendung „verborgen“ ist.

Von der $m \times n$ großen Term-Dokument Matrix A mit Rang r ausgehend wird eine Singulärwertzerlegung vorgenommen, d. h. Matrix A wird in ihre Faktoren T, Σ sowie D^T zerlegt (siehe auch Abbildung 3.5). Daraus ergibt sich Gleichung 3.1:

$$A = T\Sigma D^T \tag{3.1}$$

Dabei beschreiben die Zeilen der $m \times r$ großen, orthogonalen Matrix T die Termvektoren und werden als Links-Singulärvektoren bezeichnet, während die Zeilen der $r \times n$ großen, orthogonalen Matrix D die Dokumentvektoren repräsentieren und Rechts-Singulärvektoren genannt werden. Die $r \times r$ große Diagonalmatrix Σ enthält die korrespondierenden, nach absteigender Größe sortierten Singulärwerte von A (Martin und Berry 2007).

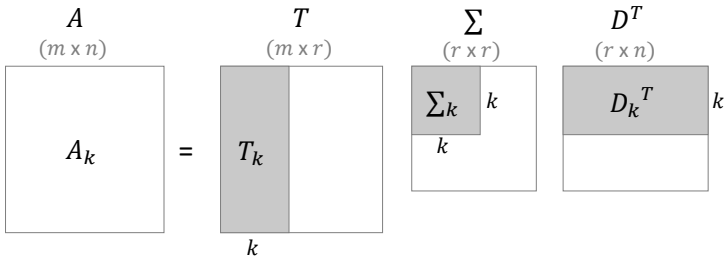


Abb. 3.5: Schematische Darstellung von Singulärwertzerlegung und Dimensionsreduktion (Martin und Berry 2007)

Die LSA verwendet zur Analyse von Bedeutungen den auf k Dimensionen verkürzten semantischen Raum. Dieser wird unter Beibehaltung der k größten Singulärwerte und den damit verbundenen Einträgen der Links- bzw. Rechts-Singulärvektoren erzeugt. Die dabei zugrunde liegende Idee ist, dass in einer Term-Dokument Matrix wichtigere und weniger wichtige Dimensionen existieren (repräsentiert durch die Größe des entsprechenden Singulärwerts) und dass die weniger wichtigen Dimensionen in gewissem Sinne Verzerrungen darstellen, die entfernt werden sollen. Das Produkt A_k der daraus entstandenen Matrizen T_k, Σ_k sowie D_k^T ist die beste (Kleinste-Quadrate) Approximation der Term-Dokument Matrix A mit k Parametern (Deerwester u. a. 1990). Dies wird durch Gleichung 3.2 verdeutlicht:

$$T_k \Sigma_k D_k^T = A_k \approx A \tag{3.2}$$

Das Ziel der Dimensionsreduktion ist die Erfassung der gemeinsamen Bedeutungen von Wörtern und Dokumenten, wodurch insbesondere die Herausforderungen durch die Verwendung von Synonymen und Polysemie bei der Auffindung ähnlicher Dokumente angegangen werden sollen (Deerwester u. a. 1990). Somit ist nach der Dimensionsreduktion ein Ähnlichkeitsvergleich auf Grundlage des semantischen Inhalts möglich und er basiert nicht mehr nur auf dem syntaktischen Vergleich von Zeichenketten (Martin und Berry 2007). Die Wirkungsweise der Dimensionsreduktion in Verbindung mit der Singulärwertzerlegung ist in Abbildung 3.6 dargestellt. Im herkömmlichen Vektorraummodell (linkes Koordinatensystem) bilden die Terme bzw. Wörter die Dimensionen des zugrunde liegenden Vektorraums. Auf Basis dieser zueinander orthogonalen Achsen können Dokumente aufgrund der Wörter, die sie enthalten, als Vektoren dargestellt werden. Aus der Orthogonalität syntaktisch ver-

schiedener Terme folgt jedoch, dass z. B. die beiden Terme „Automobil“ und „Fahrzeug“ keinerlei Ähnlichkeit aufweisen.

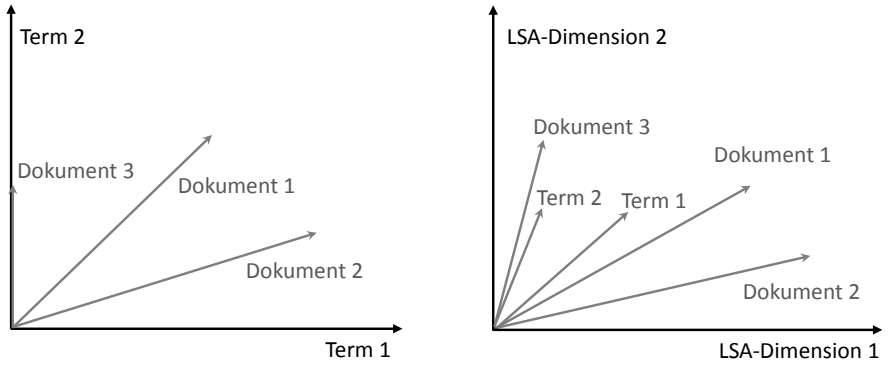


Abb. 3.6: Schematischer Vergleich der Vektorrepräsentationen des typischen Vektorraummodells (links) und des semantischen Raums der LSA (rechts) (Dumais 2007)

Bei der LSA bilden hingegen die erzeugten k Dimensionen den semantischen Raum (rechtes Koordinatensystem), in dem sowohl Dokumente als auch Terme als Vektoren abgebildet sind. Diese k Dimensionen entstehen durch Linearkombinationen der betrachteten Terme. Durch die Nutzung einer geringeren Anzahl an Dimensionen als der Anzahl einzigartiger Terme, werden Ähnlichkeiten zwischen den Termen durch die LSA aufgedeckt, insbesondere auch für Terme, die nie zusammen in einem Dokument enthalten waren (Dumais 2007). Damit werden sowohl Dokumente als auch Wörter durch Vektoren repräsentiert, deren Ähnlichkeit auf Grundlage ihrer Bedeutung quantifiziert werden kann. Für die Beispieldokumente und -terme aus Abbildung 3.6 wird es im semantischen Raum auf diese Weise möglich, *Dokument 3* über *Term 1* zu finden, obwohl *Dokument 3* den *Term 1* nicht enthält, wie die Darstellung im herkömmlichen Vektorraum zeigt. Zur eigentlichen Berechnung der LSA werden die folgenden vier Schritte ausgeführt:

1. **Erstellung einer Term-Dokument Matrix:**
   In diesem Prozessschritt werden die Terme aus den zu vergleichenden Textdokumenten extrahiert. An dieser Stelle kann es sinnvoll sein, den Rohtext mit Verfahren aus dem Natural Language Processing wie etwa zur Entfernung von Stoppwörtern oder durch den Einsatz von Stemming-Verfahren vorzuverarbei-

ten (Turney und Pantel 2010). Deerwester u. a. (1990) entfernen etwa Stoppwörter in ihrer Evaluation. Ergebnis dieses Schritts ist eine Term-Dokument Matrix gemäß Abbildung 3.3, deren Einträge absolute Häufigkeiten eines Terms in einem bestimmten Dokument darstellen.

2. **Transformation einer Term-Dokument Matrix:**
   Zur weiteren Vorverarbeitung einer Term-Dokument Matrix werden die enthaltenen absoluten Worthäufigkeiten durch lokale oder globale Gewichtungsfunktionen wie beispielsweise die Log-Entropie transformiert. Hierdurch soll das Retrieval Ergebnis verbessert werden, indem „wertvolle“ Terme, die besonders dazu geeignet sind Dokumente voneinander zu unterscheiden, höher gewichtet werden als sogenannte High-Frequency Terme. Diese High-Frequency Terme, die in vielen Dokumenten der betrachteten Dokumentenmenge vorkommen, erhalten hingegen niedrige Gewichte (Salton und Buckley 1991). Die zugrunde liegende Idee ist dabei, dass Terme, die nur in wenigen Dokumenten vorkommen, besser zur Unterscheidung und Identifikation geeignet sind als Terme, die in vielen Dokumenten enthalten sind.

3. **Singulärwertzerlegung und Dimensionsreduktion:**
   Eine Term-Dokument Matrix wird durch die Singulärwertzerlegung in drei Matrizen aufgeteilt (siehe auch Abbildung 3.5). Auf Basis der Singulärwertzerlegung einer Term-Dokument Matrix wird durch eine Dimensionsreduktion schließlich der semantische Raum gebildet. Die Dimensionsreduktion auf k Dimensionen wird unter Berücksichtigung der berechneten Singulärwerte der Term-Dokument Matrix vorgenommen. Dimensionen korrespondierender hoher Singulärwerte werden beibehalten, während die zu niedrigen Singulärwerten gehörigen Dimensionen 0 gesetzt werden. Die Folge ist, dass die ursprüngliche Vektorrepräsentation der Terme und Dokumente durch Vektoren mit nur noch k Dimensionen ersetzt wird. Dieser Schritt stellt eine Stärke der LSA dar, indem textinhärente semantische Strukturen entdeckt und redundante Informationen entfernt werden (Hu u. a. 2007), d. h. Terme bzw. Dokumente ähnlicher (konzeptioneller) Bedeutung weisen nun ähnliche Vektorrepräsentationen auf (Martin und Berry 2007). Problematisch ist hierbei allerdings die Bestimmung der optimalen Dimensionsgröße k, da keine allgemein beste Größe k existiert (Dumais 2007). Für das in Kapitel 5 beschriebene ähnlichkeitsbasierte Suchverfahren werden Lösungsmöglichkeiten in Abschnitt 5.4.3 diskutiert.

4. **Retrieval im semantischen Raum:**
   Im erzeugten semantischen Raum können die k-dimensionalen Vektorrepräsentationen der Terme und Dokumente miteinander verglichen werden. Dabei ist die Anwendung unterschiedlicher Berechnungsverfahren wie die Kosinus-Ähnlichkeit oder die Euklidische Distanz zur Quantifizierung der Ähnlichkeit zwischen Vektoren denkbar. Ergebnis dieses Schritts ist eine Maßzahl, die Auskunft darüber gibt, wie ähnlich sich zwei Objekte (Dokumente bzw. Terme) sind.

3.3.2 Ähnlichkeitsberechnungen im semantischen Raum

Grundsätzlich sind drei Arten von Ähnlichkeitsberechnungen möglich: Term-Term, Dokument-Dokument und Term-Dokument Vergleiche. Die hauptsächlich mit der LSA durchgeführten Vergleiche sind nachfolgend erläutert, wobei später in der ähnlichkeitsbasierten Suchfunktion für Prozessmodelle auf den Dokument-Dokument Vergleich zurückgegriffen wird, indem Prozessmodelle als Dokumente im Sinne der LSA repräsentiert werden (siehe Definition 5.1).

- **Term-Term Vergleich:** Um zwei Terme miteinander auf Ähnlichkeit zu vergleichen, bilden Martin und Berry (2007) das Skalarprodukt der entsprechenden Zeilen in der reduzierten Matrix A_k. Zur Generierung einer quadratischen Matrix, welche die Skalarprodukte aller möglichen Term-Term Vergleiche enthält, wird daher Formel 3.3 angewendet. Daraus lässt sich auch ableiten, dass für einen Vergleich zwischen zwei beliebigen Termen die entsprechenden zwei Zeilen i und j der Matrix $T_k\Sigma_k$ verglichen werden müssen.

$$A_k A_k^T = T_k\Sigma_k D_k^T (T_k\Sigma_k D_k^T)^T = T_k\Sigma_k D_k^T D_k\Sigma_k T_k^T = T_k\Sigma_k\Sigma_k T_k^T \tag{3.3}$$

- **Dokument-Dokument Vergleich:** Ein Ähnlichkeitsvergleich zwischen Dokumenten wird analog zum Term-Term Vergleich durchgeführt. Der Unterschied ist hierbei, dass die Spalten der reduzierten Matrix A_k miteinander verglichen werden. Formel 3.4 berechnet die quadratische Matrix aller möglichen Skalarprodukte. Daraus lässt sich analog ableiten, dass für einen Vergleich zwischen zwei beliebigen Dokumenten die entsprechenden zwei Spalten i und j der Matrix $D_k\Sigma_k$ verglichen werden müssen.

$$A_k^T A_k = (T_k\Sigma_k D_k^T)^T T_k\Sigma_k D_k^T = D_k\Sigma_k T_k^T T_k\Sigma_k D_k^T = D_k\Sigma_k\Sigma_k D_k^T \tag{3.4}$$

Darüber hinaus weisen Martin und Berry (2007) darauf hin, dass für jeden Ähnlichkeitsvergleich zwischen zwei Termen bzw. zwischen zwei Dokumenten die entsprechenden Vektoren durch die korrespondierenden Singulärwerte skaliert werden müssen. Dies sollte unabhängig von dem verwendeten Ähnlichkeitsmaß durchgeführt werden. Schließlich findet sich in (Landauer u. a. 1998) eine weiterführende Einführung zur Latent Semantic Analysis mit über die zuvor beschriebenen Ausführungen hinausgehenden Erläuterungen.

3.4 Qualitätsbestimmung von Anfrageergebnissen

Zur Bestimmung der Qualität der entwickelten Suchfunktionalitäten wird auf Evaluationsmaße aus dem Information Retrieval zurückgegriffen. Diese Maße basieren dabei auf einem Konzept von *Relevanz*. Diesem Konzept liegt die Idee zugrunde, dass zu einer Suchanfrage möglichst nur relevante Dokumente zurückgegeben werden sollen. Nicht relevante Dokumente sollten dagegen möglichst nicht im Ergebnis enthalten sein.[6] Somit ist das Konzept der Relevanz von Suchergebnissen in dem Sinne *subjektiv*, als dass für eine Anfrage *korrekte* Dokumente zur Bestimmung der Qualität benötigt werden, die häufig von Experten bestimmt werden (van Rijsbergen 1979).

Für die Evaluationen in dieser Arbeit werden die Maße *Precision*, *Recall* und *F-Wert* verwendet, die auf dem Konzept der Relevanz beruhen. Zur Definition dieser Maße werden zunächst einige Mengen für die Ergebnisse einer Anfrage eingeführt, die in Abbildung 3.7 dargestellt sind (siehe zur Definition der Maße und Mengen auch (Sokolova und Lapalme 2009)).

Die Menge aller Ergebnisse X einer Anfrage ergibt sich zum einen aus der Menge der relevanten Dokumente A und der Menge der nicht relevanten Dokumente $\overline{A}$, also $X = A + \overline{A}$. Zum anderen kann X auch für eine Anfrage in die Menge B der erhaltenen Dokumente und $\overline{B}$, der Menge der nicht erhaltenen Dokumente, aufgeteilt werden. Mit Hilfe dieser Mengen können vier verschiedene Ergebnismengen berechnet werden:

[6] Dieser Abschnitt bezieht sich auf Dokumente wie im Information Retrieval üblich. Im Kontext der Suche in Prozessmodellbibliotheken können Dokumente beim Prozessmodell-Matching als Aktivitäten aufgefasst werden, zu denen ähnliche Aktivitäten gesucht werden. Im Falle der ähnlichkeitsbasierten Suche können Dokumente als Prozessmodelle interpretiert werden, für die ähnliche Modelle gefunden werden sollen.

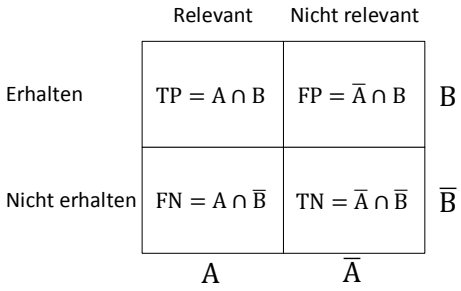


Abb. 3.7: Übersicht der zur Berechnung von Precision und Recall verwendeten Mengen

- **True Positives (TP):** Die Menge der relevanten und auch erhaltenen Dokumente wird als True Positives bezeichnet. Der Durchschnitt der Mengen A und B sollte also möglichst groß sein, da diese Menge die für einen Nutzer gesuchten, relevanten Dokumente enthält.
- **False Negatives (FN):** Die Menge der relevanten, aber nicht erhaltenen Dokumente wird als False Negatives bezeichnet. Der Durchschnitt der Mengen A und $\overline{B}$ sollte möglichst klein sein, da ein Nutzer diese für ihn relevanten Dokumente nicht erhält.
- **False Positives (FP):** Die Menge der nicht relevanten, aber erhaltenen Dokumente wird als False Positives bezeichnet. Auch diese Menge sollte möglichst klein sein, um einem Nutzer nicht zu viele für ihn nicht relevante Dokumente zu liefern.
- **True Negatives (TN):** Die Menge der nicht relevanten und auch nicht erhaltenen Dokumente wird als True Negatives bezeichnet. Diese Menge sollte wiederum möglichst groß sein, da sie widerspiegelt, wie gut für einen Nutzer nicht relevante Dokumente gefiltert werden.

Der Precision-Wert einer Anfrage lässt sich aus den beschriebenen Mengen berechnen als $P = \frac{|A \cap B|}{|B|} = \frac{|TP|}{|B|}$ und gibt das Verhältnis zwischen den relevanten, erhaltenen Dokumenten und allen erhaltenen Dokumenten an. Der Recall-Wert einer Anfrage lässt sich hingegen berechnen als $R = \frac{|A \cap B|}{|A|} = \frac{|TP|}{|A|}$ und gibt das Verhältnis der relevanten, erhaltenen Dokumente zu allen relevanten Dokumenten an. Der F-Wert kann darauf

aufbauend als gewichteter Durchschnitt von Precision und Recall interpretiert werden und berechnet sich als harmonischer Durchschnitt $F = 2 \cdot \frac{P \cdot R}{P+R} = 2 \cdot \frac{TP}{2TP+FP+FN}$.

Der F-Wert ist von besonderer Relevanz bei der Evaluation von Suchfunktionen, da sich einzeln jeweils leicht sehr hohe Precision- bzw. Recall-Werte erzielen lassen. Der Recall-Wert einer Suchanfrage beträgt etwa 1, wenn alle Dokumente in das Suchergebnis aufgenommen werden. Je nach Größe der Ergebnismenge ergibt sich daraus jedoch ein sehr geringer Precision-Wert. Durch die gemeinsame Betrachtung von Precision und Recall im F-Wert werden nur hohe Werte erzielt, wenn sowohl für Precision als auch Recall hohe Werte vorliegen.

Zum Vergleich der Suchfunktionalitäten über mehrere Anfragen hinweg, werden jeweils die zwei Durchschnittswerte *Macro-Average* und *Micro-Average* sowie die Standardabweichung eingesetzt. Der Macro-Average Wert berechnet sich dabei als Durchschnitt von Precision, Recall und F-Wert über alle Anfragen hinweg. Demgegenüber werden zur Berechnung des Micro-Average Werts die Werte für TP, TN, FP und FN über alle Anfragen hinweg aufsummiert und anschließend Precision, Recall und F-Wert berechnet. Zur Evaluation werden beide Durchschnittsberechnungen angegeben, da sich die Anzahl der korrekten Ergebnisse für verschiedene Anfragen unterscheidet. Der Micro-Average Wert gewichtet durch das Aufsummieren von TP, TN, FP und FN jedes Ergebnis einer Anfrage gleich, sodass unterschiedliche Ergebnisgrößen keinen Einfluss auf den Durchschnitt haben. Der Macro-Average Wert betrachtet hingegen das Ergebnis einer Anfrage als Ganzes. So hat beispielsweise der Precision-Wert einer Anfrage mit kleiner Ergebnisgröße den gleichen Einfluss auf den Macro-Average, wie eine Anfrage mit großer Ergebnisgröße.[7]

Für die in Kapitel 4 erläuterten Verfahren werden zudem die von Kuss u. a. (2016) entwickelten Evaluationsmaße *Probabilistische Precision* (ProP), *Probabilistischer Recall* (ProR) und *Probabilistischer F-Wert* (ProFW) verwendet. Im Prozessmodell-Matching Bereich werden die True Positives typischerweise durch einen von Experten erstellten sogenannten *Goldstandard* festgelegt (siehe etwa die Beschreibungen der Matching Wettbewerbe (Cayoglu u. a. 2014; Antunes u. a. 2015)). Da die Bestimmung eines eindeutigen Standards von Matches aufgrund unterschiedlicher möglicher Interpretationen von Aktivitäten jedoch nicht möglich ist (Thaler u. a. 2014), schlagen Kuss u. a.

[7] Vergleiche dazu auch die Ausführungen in (Lipton u. a. 2014) im Kontext von Klassifikationsfragestellungen.

(2016) eine Kombination mehrerer Standards und deren gemeinsame Evaluation vor. Dazu definieren sie die folgenden Kennzahlen:

$$ProP = \frac{sup(TP)}{sup(TP) + sup_{max} \cdot (|\Lambda \setminus C|)} \tag{3.5}$$

$$ProR = \frac{sup(TP)}{sup(A)} \tag{3.6}$$

$$ProFW = 2 \cdot \frac{ProP \cdot ProR}{ProP + ProR} \tag{3.7}$$

Λ bezeichnet die Menge an Matches, die durch ein Matching Verfahren ermittelt wurde (siehe auch Definition 4.2). C enthält die Menge an Matches, die sich aus der Zusammenfassung aller Goldstandards ergibt. Die probabilistische Precision ergibt sich dabei als Anteil der Unterstützung aller True Positives ($sup(TP)$) von der Unterstützung aller True Positives plus zusätzlich der maximalen Unterstützung der Anzahl von Matches, die zwar von einem automatisierten Matching Verfahren bestimmt wurden, aber nicht in einem Goldstandard enthalten sind ($sup_{max} \cdot (|\Lambda \setminus C|)$). Unter dem Begriff *Unterstützung* ist in den obigen Formeln die Häufigkeit eines korrekten Matches in den Standards zu verstehen. Sollte z. B. ein $\lambda_i \in \Lambda$ Bestandteil von drei Standards sein, so beträgt der entsprechende Unterstützungswert ebenso drei. $sup_{max} \cdot (|\Lambda \setminus C|)$ weist jedem automatisierten Match, das nicht in einem Standard enthalten ist (False Positive), den maximalen Unterstützungswert zu.

Da nicht klar ist, wie hoch die Unterstützung eines False Positive ist, muss ein Wert festgelegt werden. In (Kuss u. a. 2016) wird dazu der maximale Unterstützungswert gewählt, was zu einer maximalen Bestrafung von False Positives bei der Berechnung von ProP führt. Somit wird der Fokus auf die Vermeidung von False Positives gelegt. Davon wird in der Evaluation in Kapitel 4 abgewichen, um eine ausgewogene Gewichtung zu erzielen und den Fokus in der Evaluation nicht auf die Vermeidung von False Positives zu legen. Anstatt der maximalen Unterstützung wird daher der Median verwendet. Dadurch würde im Beispiel von oben zur Gewichtung von False Positives als Unterstützungswert zwei verwendet und nicht die maximale Unterstützung von drei. Probabilistischer Recall wird als Anteil der Unterstützung aller True Positives von der Unterstützung aller relevanten Matches angesehen. Und der probabilistische F-Wert ergibt sich analog zum F-Wert als harmonisches Mittel aus probabilistischer Precision und probabilistischem Recall.

Diese Kennzahlen werden insbesondere für die Evaluation der Prozessmodell-Matching Verfahren eingesetzt, da sich Goldstandards verschiedener Experten deutlich voneinander unterscheiden. So stimmen die von vier Experten erstellten Goldstandards für die Untersuchungen in (Rodriguez u. a. 2016) lediglich zu maximal 60,9% überein. Die in (Kuss u. a. 2016) in den Experimenten verwendeten Standards unterscheiden sich ebenfalls deutlich. Die acht Experten identifizierten insgesamt 879 Matches, wobei allerdings nur 495 Matches von mindestens vier Experten genannt wurden. Diese Beobachtung wird auch durch die Evaluation in Kapitel 4 bestätigt. Für den einen verwendeten Datensatz erkannten die fünf befragten Experten beispielsweise insgesamt 651 Matches, wohingegen nur 311 Matches von mindestens drei Experten identifiziert wurden (siehe auch Kapitel 4.4). Daraus wird ersichtlich, dass die Erstellung eines Goldstandards eine herausfordernde Aufgabe darstellt. Ein konkretes Beispiel bei der Erstellung der Standards für die Evaluation in Kapitel 4.4 betrifft etwa die Beurteilung von Auswahlgesprächen und Bewerbungstests im Rahmen von Zulassungsprozessen an Universitäten. In diesem Zusammenhang wurden Aktivitäten bezüglich Auswahlgesprächen und Bewerbungstests von einigen Experten als Match aufgefasst, von anderen jedoch nicht. In (Thaler u. a. 2014) finden sich zudem weitere illustrative Beispiele, die verdeutlichen, dass durchaus – in Abhängigkeit der Zielstellung eines Matchings und der persönlichen Präferenzen von Experten – mehrere sinnvolle Varianten eines Goldstandards existieren können.

Zusätzlich werden für das in Kapitel 5 beschriebene ähnlichkeitsbasierte Suchverfahren für Prozessmodelle noch zwei weitere Qualitätsmaße berechnet. *Precision-at-k*[8] berechnet Precision basierend auf einer nach Ähnlichkeitswerten absteigend sortierten Reihenfolge von Modellen, wobei zur Berechnung nur die ersten k Ergebnisse herangezogen werden. Es wird somit ermittelt, wie groß der Anteil an True Positives unter den k-ersten Modellen in einer Reihenfolge ist. *R-Precision* stellt eine mit Precision-at-k eng verwandte Maßzahl dar. Dabei wird k nicht beliebig gewählt, sondern es wird Precision für die Anzahl an relevanten Modellen A berechnet. Sind beispielsweise für eine Anfrage 10 Modelle relevant, in der Ergebnisliste aber 20 Modelle vorhanden, so wird Precision für die 10 Modelle mit den höchsten Ähnlichkeitswerten bestimmt. Die Modelle werden also absteigend nach Ähnlichkeitswert sortiert. Für weitere Erläuterungen zu diesen Kennzahlen wird auf (Manning u. a. 2008) verwiesen.

[8] Zur eindeutigen Bezeichnung von Precision-at-k wird in Kapitel 5 Precision-at-n verwendet, da sowohl im Kontext der LSA die Dimensionszahl des semantischen Raums und für die Kennzahl Precision-at-k die Variable k verwendet wird.

3.5 Zusammenfassung

In diesem Kapitel wurden zunächst grundlegende Techniken zur Verarbeitung natürlicher Sprache eingeführt, die in den Suchfunktionen dazu verwendet werden die Beschriftungen von Prozessmodellelementen vor der eigentlichen Ähnlichkeitsberechnung vorzuverarbeiten. Dazu gehören die Zerlegung von Wortfolgen in einzelne Wörter sowie die Reduktion von Wörtern auf ihre Stammform. Zudem wurden Stoppwörter als eine Klasse von Wörtern vorgestellt, die wenig zur eigentlichen Semantik einer Beschriftung beitragen und die daher typischerweise vor der Ergebnisberechnung entfernt werden.

Zur Bestimmung der Ähnlichkeit von Wörtern wurden daraufhin drei unterschiedliche Herangehensweisen vorgestellt. Verfahren aus dem ersten Bereich betrachten Wörter als Folge von Zeichen und berechnen einen Ähnlichkeitswert basierend auf der Anzahl unterschiedlicher Zeichen. Ansätze aus dem zweiten Bereich verwenden hingegen Beziehungen zwischen Wörtern in lexikalischen Datenbanken, um eine sogenannte semantische Ähnlichkeit zu bestimmen. Schließlich nutzt eine dritte Gruppe von Verfahren große Textmengen und statistische Berechnungen um einen Ähnlichkeitswert von Wörtern zu bestimmen.

Anschließend wurde die Latent Semantic Analysis beschrieben, die im Information Retrieval für die Suche nach Dokumenten entwickelt wurde. Sie basiert auf einem Vektorraummodell, in dem sowohl Dokumente als auch Terme als Vektoren dargestellt werden und in dem die Ähnlichkeit von Dokumenten beispielsweise durch die Kosinus-Ähnlichkeit der entsprechenden Vektoren bestimmt werden kann. Die Latent Semantic Analysis wird in Kapitel 5.3 für die Suche nach ähnlichen Prozessmodellen adaptiert. Schließlich wurden Qualitätskriterien für die Evaluation der Suchfunktionalitäten eingeführt. Zu den verwendeten Maßen zählen Precision, Recall und F-Wert sowie Precision-at-k und R-Precision.

Teil II
Suchmöglichkeiten für Geschäftsprozessmodelle

Kapitel 4

Prozessmodell-Matching von Aktivitäten

Prozessmodell-Matching Techniken werden dazu eingesetzt ähnliche Aktivitäten in Prozessmodellen zu identifizieren (genereller auch andere ähnliche Modellelemente). Aktivitäten werden dabei als ähnlich betrachtet, wenn sie semantisch übereinstimmen, d. h., sie beschreiben die gleiche Aktivität in der realen Welt. Matching-Verfahren werden vielfach als Grundlage zur Bestimmung der Ähnlichkeit von Prozessmodellen verwendet (siehe etwa die Übersichten in (Schoknecht u. a. 2017b; Becker und Laue 2012) oder Kapitel 5.2). In dieser Arbeit werden sie allerdings zum einen eigenständig beschrieben, um zu bestimmen, welche Aktivitäten verschiedener Prozessmodelle zueinander ähnlich sind. Folglich kann nicht nur die Ähnlichkeit vollständiger Modelle analysiert werden, sondern auch die Übereinstimmung einzelner Modellelemente, um diese z. B. zu vereinheitlichen (Beschriftungen angleichen, Ersetzung von Aktivitäten, etc.). Zum anderen wird in Kapitel 6.3 ein Ansatz erläutert, durch den das Prozessmodell-Matching mit einer Anfragesprache für Prozessmodelle kombiniert wird. Dadurch können beispielsweise zu einer Anfrage passende Modelle gefunden werden, die eine andere Wortwahl (z. B. Synonyme) aufweisen.

In diesem Kapitel erfolgt zunächst eine Problembeschreibung des Prozessmodell-Matchings (Kapitel 4.1), woraufhin auf existierende Ansätze eingegangen wird (Kapitel 4.2). Schließlich werden in Kapitel 4.3 zwei Verfahren für das Matching von

Prozessmodellen vorgestellt. Diese Ansätze sind darauf ausgerichtet ähnliche Transitionen innerhalb von zwei als Petri-Netz modellierten Prozessmodellen zu identifizieren. Auf die Implementierung der Ansätze und eine Evaluation wird in Abschnitt 4.4 vertieft eingegangen.

4.1 Problemdefinition und Beispiel

Im Allgemeinen wird unter *Matching* ein Verfahren verstanden, das zwei Modelle als Eingabe erhält, als Quelle und Ziel bezeichnet, und daraufhin eine Anzahl von *Matches* zwischen den beiden Modellen berechnet. Diese Matches basieren auf einem gewissen Maß an Entsprechung (Rahm und Bernstein 2001) und repräsentieren eine Beziehung zwischen der Potenzmenge der Knoten des Quellmodells und der Potenzmenge der Knoten des Zielmodells. Die Semantik von sich entsprechenden Knoten stimmt somit für Menschen in gewisser Weise überein. Dabei kann der Begriff Modell weitläufig aufgefasst werden und umfasst etwa Datenbankschemata (siehe z. B. (Evermann 2009)), Ontologien (Euzenat und Shvaiko 2013) oder Prozessmodelle.

Unter dem Begriff *Prozessmodell-Matching* wird im Speziellen das Matching von einzelnen Knoten oder von Mengen von Knoten eines Prozessmodells auf entsprechende Knoten eines anderen Modells verstanden (Cayoglu u. a. 2014). Ein solches Matching basiert auf Kriterien wie z. B. Ähnlichkeit, Äquivalenz oder Analogie (Thaler u. a. 2014). Im Wesentlichen soll durch ein Match[1] ausgesagt werden, dass die darin enthaltenen Aktivitäten die gleichen Tätigkeiten innerhalb einer Organisation repräsentieren. Im Folgenden wird diese Idee für die Abbildung von Transitionen zwischen zwei beschrifteten Workflow-Netzen BWN_1 und BWN_2 betrachtet, die jeweils einen Geschäftsprozess darstellen.

Formal beschrieben berechnen die in Kapitel 4.3 vorgestellten Ansätze daher Matches, die folgender Relation $match : \mathcal{P}(T_1) \times \mathcal{P}(T_2)$ entsprechen. Ein *Match* $(T_i, T_j) \in match, T_i \subseteq T_1, T_j \subseteq T_2$ beschreibt somit, dass die Menge an Transitionen T_i aus BWN_1 der Menge an Transitionen T_j aus BWN_2 entspricht. Zur Bestimmung von Matches wird typischerweise ein Ähnlichkeitswert zwischen T_i und T_j berechnet. Sollte dieser Ähnlichkeitswert über einem Schwellenwert θ liegen, werden T_i und T_j als Match betrachtet.

[1] In der Literatur werden diese teilweise auch als *Correspondences* (Weidlich u. a. 2010) bezeichnet.

Ein *Matching*[2] zwischen zwei Prozessmodellen P_1 und P_2 mit zugehörigen Transitionsmengen T_1 und T_2 entspricht damit letztlich der Menge an Matches zwischen diesen Modellen:

Definition 4.1: Match/Matching

Die Menge an Matches Λ von zwei beschrifteten Workflow-Netzen BWN_1 und BWN_2 enthält Tupel (T_i, T_j), deren Ähnlichkeitswert mindestens so groß wie ein Schwellenwert θ ist: $\Lambda = \{(T_i, T_j) \mid Sim(T_i, T_j) \geq \theta, T_i \subseteq T_1, T_j \subseteq T_2\}$. Ein Matching von zwei beschrifteten Workflow-Netzen entspricht dann der Menge an Matches: $matching(BWN_1, BWN_2) := \Lambda$.

Bei T_1 und T_2 handelt es sich um die Mengen von Transitionen der als beschriftete Workflow-Netze dargestellten Geschäftsprozessmodelle BWN_1 und BWN_2. Sinnvoller Weise gilt dabei $T_1 \neq \emptyset$ und $T_2 \neq \emptyset$, da sich ansonsten keine Matches berechnen lassen. Diese auf Aktivitäten bzw. Transitionen beschränkten Definitionen lassen sich ersichtlich auch auf andere Modellierungskonstrukte wie etwa Stellen übertragen, indem eine entsprechende $match$ Relation definiert wird.

Das Resultat der Anwendung von Prozessmodell-Matching Techniken ist somit eine Menge von Matches. Dabei wird generell zwischen elementaren und komplexen Matches differenziert, was folgende Definition beschreibt:

Definition 4.2: Elementares / Komplexes Match

Sei Λ eine Menge von Matches. Dann ist ein Match $\lambda \in \Lambda$ ein Tupel (T_i, T_j) zweier Mengen von Transitionen der beschrifteten Workflow-Netze BWN_1 und BWN_2. Ein Match (T_i, T_j) wird als elementar bezeichnet, wenn $|T_i| = |T_j| = 1$ und als komplex, wenn $|T_i| > 1 \vee |T_j| > 1$ (in Anlehnung an (Weidlich u. a. 2010)).

In Bezug auf die tatsächliche Berechnung solcher Matches ist damit allerdings noch nichts ausgesagt. Im Allgemeinen können die in Kapitel 2.2.2 vorgestellten Dimensionen zur Berechnung von Matches berücksichtigt werden. Hier erscheint zunächst vor

[2] Teilweise wird ein Matching in der Literatur auch als *Alignment* (Dijkman u. a. 2009a) oder als *Mapping* (Antunes u. a. 2015) bezeichnet.

allem die Dimension der natürlichen Sprache sinnvoll, da über die Beschriftungen von Prozessmodellelementen der Inhalt des Modells sowie der zugrunde liegende Prozess beschrieben werden. Diese Dimension wird auch stark von bisher veröffentlichten Ansätzen berücksichtigt, wohingegen die Crowd-Dimension bislang kaum eine Rolle spielt. Auch die Struktur bzw. das Verhalten eines Modells werden häufiger für die Berechnung von Matches eingesetzt. Dabei ist der Einsatz „simpler" Ansätze wie beispielsweise der Berechnung der Levenshtein-Distanz (Levenshtein 1966) zwischen unterschiedlichen Beschriftungen oder die Anzahl von ein- und ausgehenden Kanten hinsichtlich der Graphstruktur allein typischerweise nicht ausreichend, was folgendes Beispiel illustriert.

In Abbildung 4.1 sind zwei Prozessmodelle abgebildet, die jeweils einen Bestellvorgang repräsentieren. Die beiden Modelle entsprechen sich nicht vollständig, so unterscheiden sich teilweise die Beschriftungen der Transitionen in der Wortwahl und der Versand der Ware ist unterschiedlich detailliert modelliert. Die grün umrandeten Bestandteile stellen exemplarisch verschiedene Matches dar. Zum einen entsprechen sich die Transitionen „Bestellung anlegen" und „Auftrag anlegen" aus den Modellen a) und b). Diese Transitionen unterscheiden sich nur durch die synonymen Substantive und stellen ein *elementares* Match dar. Zum anderen entspricht die Transition „Ware versenden" aus Modell b) den Transitionen „Ware verpacken" und „Ware versenden" aus Modell a). Dieser Teil von Modell a) ist in einem anderen Detaillierungsgrad modelliert als der entsprechende Teil von Modell b), wodurch sich ein *komplexes* $1 : n$ Match ergibt.

Anhand dieser Modelle lassen sich einige Herausforderungen des Prozessmodell-Matchings erkennen. Zunächst kann nicht vorausgesetzt werden, dass die Beschriftungen von Transitionen jeweils die gleiche Wortwahl aufweisen. So findet üblicherweise eine Verwendung von Synonymen, Homonymen, Antonymen, Verneinungen, etc. statt. Aus diesem Grund ist auch der alleinige Einsatz der Levenshtein-Distanz von Beschriftungen nur begrenzt sinnvoll, denn für das elementare Match in Abbildung 4.1 würde eine große Distanz und folglich eine geringe Ähnlichkeit berechnet werden. Zudem werden im Folgenden Einschränkungen wie z. B. auf die Verwendung von Richtlinien zur Beschriftung von Transitionen (Mendling u. a. 2010), die Verwendung von Satzschablonen wie in (Caporale 2016) angedacht oder die Annahme einer einheitlichen Wortwahl nicht getroffen. Eine solche „kontrollierte" Modellie-

rung (Thaler u. a. 2017) erscheint in der Praxis schwer durchsetzbar.[3] Die beschriebenen Matching-Ansätze versuchen solche Herausforderungen durch den Einsatz von Techniken aus dem Bereich des Natural Language Processing zu lösen.

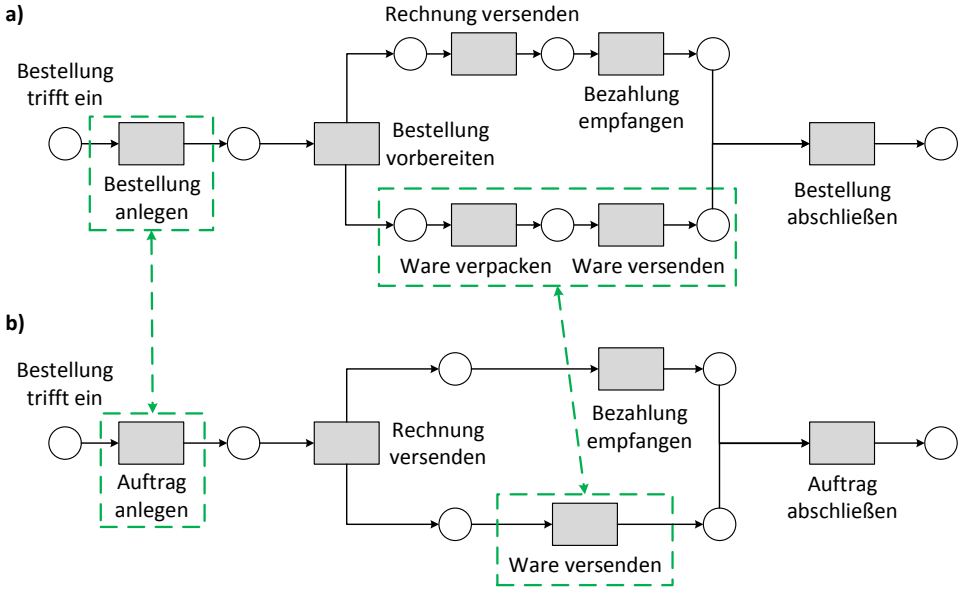


Abb. 4.1: Beispiele für Matches in Prozessmodellen

Das abgebildete $1 : n$ Match zeigt eine weitere Herausforderung, die sich durch die unterschiedliche Detaillierung bei der Modellierung ergibt. Hier besteht die Schwierigkeit darin, zu erkennen, dass sich nicht nur die Transitionen „Ware versenden“ entsprechen, sondern dass ebenso die Transition „Ware verpacken“ aus Modell a) Bestandteil dieses Matches ist. Auch bezüglich des Detaillierungsgrads von Modellen werden im Folgenden keine Einschränkungen gemacht, da es vor allem schwierig erscheint festzulegen, wann genau ein Modell zu einem anderen Modell einen unterschiedlichen Detaillierungsgrad aufweist oder z. B. nur andere Aktivitäten beinhaltet. Diesen Aspekt versuchen die Triple-S Ansätze durch eine Berücksichtigung der Struktur von Prozessmodellen zu lösen.

[3] Siehe dazu auch die Erläuterungen zu Beschriftungen in Kapitel 2.1.1.

4.2 Existierende Ansätze zum Prozessmodell-Matching

Für die Suche nach verwandten Arbeiten zum Prozessmodell-Matching sowie zur ähnlichkeitsbasierten Suche (siehe Kapitel 5.2) wurde auf die Literatursuche von Schoknecht u. a. (2017b) zurückgegriffen, die ein strukturiertes Suchvorgehen nach Webster und Watson (2002) angewendet haben. Dazu wurden zunächst die Literaturdatenbanken SPRINGER LINK, ACM, IEEE, ISI WEB OF KNOWLEDGE, EBSCO HOST und GOOGLE SCHOLAR durchsucht. In einem zweiten Schritt wurde eine Rückwärtssuche anhand der zitierten Literatur in den gefunden Veröffentlichungen durchgeführt. Im Allgemeinen wurde, sofern von der Literaturdatenbank unterstützt, der Volltext durchsucht. In allen anderen Fällen wurde die Suche auf Titel, Abstract und Schlüsselwörter begrenzt. Die Suche selbst fand im April 2016 statt, wobei spätere thematisch passende Veröffentlichungen, auf die die Autoren aufmerksam wurden, zusätzlich aufgenommen wurden. Des Weiteren wurden nach der finalen Einreichung von (Schoknecht u. a. 2017b) publizierte Beiträge im Folgenden berücksichtigt, insoweit der Autor dieser Arbeit von diesen Kenntnis erlangte.

Die folgenden Anfragen wurden an die Literaturdatenbanken gestellt: “process model” AND “matching”, “process matching”, “process model” AND “similarity”, “process similarity”, “process model” AND “duplicate”, “process model” AND “equivalence”, “identical process models”, “distance” AND “process models”, “dissimilarity” AND “process models”, “process model” AND “clustering”, “process correspondence”, “workflow matching”, “workflow similarity”, “process model” AND “comparison”, “workflow” AND “comparison”.[4]

Diese Anfragen enthalten übliche Begriffe und Synonyme im Zusammenhang mit Prozessmodell-Matching sowie der Ähnlichkeit von Prozessmodellen. Dabei wurde insbesondere neben dem eigentlichen Matching von Prozessmodellen auch nach Veröffentlichungen gesucht, die allgemein die Ähnlichkeit von Prozessmodellen untersuchen, da die Begriffe Matching und Ähnlichkeit (Similarity) oftmals im gleichen Kontext verwendet werden. Für den in diesem Kapitel im Speziellen betrachteten Fall des Prozessmodell-Matchings hat sich erst durch die beiden Wettbewerbe (Antunes u. a.

[4] Die Suchbegriffe wurden für die vorliegende Arbeit auch auf Deutsch übersetzt, jedoch wurden dadurch keine weiteren relevanten Publikationen gegenüber den englischen Anfragen gefunden, sodass von einer Auflistung hier abgesehen wird.

2015; Cayoglu u. a. 2014) eine einheitliche Begriffswahl ergeben. Dies führt dazu, dass für die verwandten Arbeiten zur im nächsten Kapitel beschriebenen ähnlichkeitsbasierten Suche das Ergebnis dieser Literatursuche ebenfalls verwendet wird. So konnte zu diesem Aspekt auch relevante Literatur gefunden werden, die nicht den Begriff Ähnlichkeit verwendet wie z. B. (Wombacher u. a. 2004). Eine Übersicht über die Trefferanzahl pro Datenbank und Suchanfrage ist in Tabelle 4.1 abgebildet.

Tab. 4.1: Trefferanzahl der Literatursuche nach (Schoknecht u. a. 2017b)

Suchbegriff	Springer	ACM	IEEE	Ebsco	ISI	Google
“process model” ∧ “matching”	19.627	698	8.324	3.817	202	51.600
“process matching”	1.046	62	338	298	24	3.410
“process model” ∧ “similarity”	11.914	442	3.292	7.316	145	42.800
“process similarity”	515	49	211	186	43	1.920
“process model” ∧ “duplicate”	2.882	145	1.067	604	7	10.400
“process model” ∧ “equivalence”	3.994	179	1.228	1.388	38	16.500
“identical process models”	7	1	2	0	0	17
“distance” ∧ “process models”	15.441	483	2.595	3.393	62	43.700
“dissimilarity” ∧ “process models”	682	13	89	263	1	1.780
“process model” ∧ “clustering”	10.156	309	3.847	1.581	71	23.100
“process correspondence”	227	4	22	524	5	1.640
“workflow matching”	31	5	11	9	1	127
“process model” ∧ “comparison”	37.701	71	13.484	11	9	715.000
“workflow” ∧ “comparison”	29.887	75	12.615	35	37	19.500.000

Da die Anzahl an Suchergebnissen teilweise sehr hoch war, wurde die Untersuchung der Literatur nach drei Ergebnisseiten ohne relevante Veröffentlichung für Suchergebnisse mit mehr als 250 Treffern abgebrochen. Die Suchergebnisse waren dabei jeweils nach der Relevanzberechnung der verwendeten Literaturdatenbanken sortiert. Somit wurden bei jeder Anfrage mindestens 30 nicht relevante, aufeinander folgende Treffer untersucht, bevor eine Suche beendet wurde. Schließlich resultierten insgesamt 191 relevante Treffer aus der Literatursuche, wobei einige nicht berücksichtigt wurden, da es sich z. B. um Literaturübersichten handelte oder ein bereits publiziertes Ähnlichkeitsmaß in einem anderen Kontext beschrieben wurde. Für das Prozessmodell-Matching sind schließlich insbesondere drei Publikationen relevant. Dabei handelt es sich um die beiden Prozessmodell-Matching Wettbewerbe (Antunes u. a. 2015; Cayoglu u. a. 2014) sowie eine weitere Veröffentlichung, die explizit verschiedene Prozessmodell-Matching Komponenten beschreibt (Weidlich u. a. 2010). Schließlich wurden

21 Publikationen mit relevanten Arbeiten zum Prozessmodell-Matching identifiziert, die im Folgenden genauer erläutert werden.

Weiterhin anzumerken ist, dass Ansätze zur Berechnung der Ähnlichkeit von Prozessmodellen, die auf einem Prozessmodell-Matching basieren, in diesem Kapitel nicht explizit berücksichtigt werden. In diesen Arbeiten liegt der Fokus auf der Berechnung einer Gesamtähnlichkeit von Modellen und nicht auf der Ähnlichkeitsberechnung zwischen Aktivitäten in Prozessmodellen. Sie stellen somit verwandte Arbeiten für die im nächsten Kapitel beschriebene ähnlichkeitsbasierte Suche dar.

Ein erster Ansatz für das Auffinden von Matches wird in (Küster u. a. 2008) erwähnt. Dabei werden sich entsprechende Aktivitäten in verschiedenen Modellen mittels eines eindeutigen Identifizierers ermittelt. Dies erscheint im Rahmen des beschriebenen Anwendungsfalls zur Differenzerkennung zwischen Prozessmodellen innerhalb eines verwendeten Systems auch umsetzbar, für den in dieser Arbeit behandelten Anwendungsfall einer Suche jedoch zu einschränkend. Zumal nichts darüber ausgesagt wird, wie diese eindeutigen Identifizierer vergeben werden bzw. wie die Korrektheit sichergestellt wird.

Zwei weitere frühe Ansätze wurden von Dijkman u. a. (2009a) beschrieben. Der erste Ansatz basiert auf einer String-Editier Distanz, wobei zwei Aktivitäten ein Match ergeben, wenn ihr Ähnlichkeitswert über einem Schwellenwert liegt. Eine weitere Variante zu diesem Ansatz nutzt bereits Techniken der natürlichen Sprachverarbeitung (Wortstammreduktion, Entfernung von Stoppwörtern), allerdings werden keine semantischen Ähnlichkeitsmaße zwischen Wörtern wie bei den Triple-S Verfahren verwendet. Als zweiter Ansatz werden zwei Varianten vorgestellt, die zudem die Struktur eines Graphen im Sinne einer Graph-Editier Distanz berücksichtigen. Die Triple-S Ansätze vergleichen hingegen die Position von Aktivitäten in den Modellen. Der Ansatz aus (Dijkman u. a. 2009a) wird im Wesentlichen auch in (La Rosa u. a. 2010; La Rosa u. a. 2013) für das Zusammenfügen von Prozessmodellen verwendet. Neben der lexikalischen Ähnlichkeit von Beschriftungen wird dabei auch ein linguistisches Ähnlichkeitsmaß (Pedersen u. a. 2004) in (La Rosa u. a. 2013) eingesetzt und das Maximum der beiden Werte zur Bestimmung eines Matches verwendet.

Niedermann u. a. (2010) verwenden neben der Levenshtein-Distanz auch die Anzahl an synonymen Wörtern in Beschriftungen von Aktivitäten, um deren Ähnlichkeit zu bestimmen. Zudem wird diese Berechnungsweise für Beschriftungen von Datenobjekten verwendet. Unter welchen Bedingungen diese Berechnungen zur Klassifikati-

on von Aktivitäten als Match führen, wird allerdings nicht exakt spezifiziert. Für die linguistische Analyse wird allerdings kein semantisches Ähnlichkeitsmaß für Wörter wie bei den Triple-S Verfahren eingesetzt, sondern lediglich die Anzahl an synonymen Wörtern betrachtet. Zudem werden keine strukturellen Aspekte für die Identifikation von Matches berücksichtigt.

In (Weidlich u. a. 2010) wird ein Framework zur Entwicklung sogenannter *Matcher* beschrieben. Diese Matcher entsprechen im Wesentlichen den in dieser Arbeit als Prozessmodell-Matching Ansatz beschriebenen Verfahren und können ebenso dazu genutzt werden, sich entsprechende Knoten in Prozessmodellen zu identifizieren. Neben dem Framework selbst werden verschiedene Ansätze zum Auffinden von Matches beschrieben. Um elementare Matches zu identifizieren, wird die Levenshtein-Distanz der Beschriftungen von Aktivitäten eingesetzt. Für komplexe $1 : n$ Matches werden hingegen zwei Ansätze verwendet, um die Aktivitäten eines Modells basierend auf dessen Struktur zu gruppieren und anschließend die Ähnlichkeit zu einer Aktivität eines anderen Modells zu bestimmen.

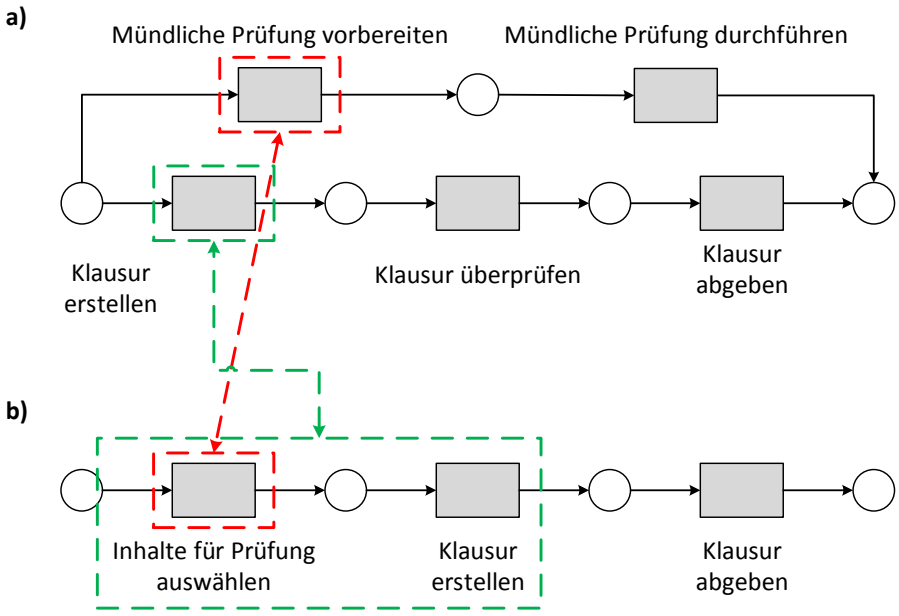


Abb. 4.2: Beispiel für sich überlappende Matches

Allerdings werden in (Weidlich u. a. 2010) überlappende Matches ausgeschlossen, eine Aktivität kann also beispielsweise nicht Bestandteil mehrerer $1 : n$ Matches sein. Diese Beschränkung ist in der vorliegenden Arbeit nicht gegeben, da es durchaus sinnvolle überlappende Matches geben kann. In Abbildung 4.2 ist dazu ein Beispiel dargestellt. In diesen Prozessmodellen ist jeweils verkürzt die Vorbereitung von Prüfungen beschrieben und die umrandeten Teile stellen sich überlappende Matches dar. In Modell b) werden dabei zunächst die Inhalte einer Prüfung ausgewählt, bevor eine Klausur erstellt und schließlich abgegeben wird. Dabei kann die Transition „Klausur erstellen“ aus Modell a) als übereinstimmend mit den Transitionen „Inhalte für Prüfung auswählen“ und „Klausur erstellen“ angesehen werden, da die Transition in a) nur in einem anderen Detaillierungsgrad modelliert wurde und ebenfalls die Auswahl des Prüfungsstoffs beinhaltet. Darüber hinaus kann die Transition „Mündliche Prüfung vorbereiten“ aus Modell a) als ein Match mit der Transition „Inhalte für Prüfung auswählen“ angesehen werden, wenn zur Vorbereitung ebenfalls die Inhaltsauswahl gehört. Da es sich jedoch nicht um eine Klausur handelt, ist die Transition „Klausur erstellen“ aus Modell b) nicht Teil des Matches.

Branco u. a. (2012) haben ebenso einen Matching-Ansatz für das Auffinden von sich entsprechenden Aktivitäten in Prozessmodellen mit unterschiedlichem Abstraktionsgrad entwickelt. Das Matching-Vorgehen selbst besteht aus zwei Phasen: Zunächst werden elementare Matches durch einen Vergleich von Attributen der Modellelemente ermittelt, um anschließend komplexe Matches mit Hilfe einer Process Structure Tree (PST) Darstellung (Vanhatalo u. a. 2007) von Prozessmodellen zu finden. Die überprüften Attribute sind dabei der Typ und die Bezeichnung von Modellelementen, die auf ihre Gleichheit hin getestet werden. Für komplexe Matches werden demgegenüber Regionen in PSTs anhand der Ähnlichkeit von Bigrammen der Typen und Bezeichnungen verglichen. Die Triple-S Ansätze unterscheiden sich von (Weidlich u. a. 2010) und (Branco u. a. 2012), indem sie versuchen auch die unterschiedliche Wortwahl bei der Modellierung durch den Einsatz eines semantischen Ähnlichkeitsmaßes für Wörter zu berücksichtigen. Die Struktur eines Modells wird zusätzlich durch die Position von Aktivitäten verglichen, ohne diese zu gruppieren.

Ein weiterer Ansatz nutzt Prozessmodell-Matching, um einem Modellierer während der Modellierung Vorschläge zu unterbreiten (Chan u. a. 2012). Dabei wird ebenfalls die Levenshtein-Distanz zur Berechnung der Ähnlichkeit von Modellelementen genutzt. Zusätzlich werden die über Kanten verbundenen Elemente, der sogenannte Kontext, berücksichtigt, indem die gewichtete Ähnlichkeit dieser Elemente mit in die

Berechnung einfließt. Schließlich wird dem Modellierer eine gewisse Anzahl an Modellelementen vorgeschlagen, die die höchsten Ähnlichkeitswerte aufweisen. Becker u. a. (2012) verwenden hingegen neben den Beschriftungen von Aktivitäten die Verhaltensdimension, um Matches zu berechnen. Die Ähnlichkeit von Beschriftungen wird als lokale Ähnlichkeit aufgefasst, während das Ablaufverhalten eines Modells als globale Ähnlichkeit betrachtet wird. Dabei wird untersucht, inwieweit ein Modell das Verhalten eines anderen Modells simulieren kann, was als Lineares Optimierungsproblem dargestellt wird, durch das lokale Ähnlichkeitswerte beeinflusst werden.

Zwei Matching-Ansätze, die explizit versuchen mit der sprachlichen Heterogenität verschiedener Modelle umzugehen, sind in (Leopold u. a. 2012; Weidlich u. a. 2013) beschrieben. Leopold u. a. (2012) beschreiben ein zweistufiges Vorgehen, bei dem zunächst die Semantik von Beschriftungen erfasst wird. Anschließend werden mit Hilfe probabilistischer Optimierung basierend auf der Reihenfolgenbeziehung von Aktivitäten Matches ermittelt. Zur Erfassung der Semantik von Beschriftungen werden diese in drei Teile zerlegt, um festzulegen welcher Teil der Aktion, dem Geschäftsobjekt oder einem sonstigen Fragment entspricht. Diese Beschriftungsteile werden anschließend durch das semantische Ähnlichkeitsmaß für Wörter von Lin (1998) auf ihre Ähnlichkeit hin untersucht. Dahingehend ist dieser Ansatz verwandt zur ersten Triple-S Variante, die zur Bestimmung der semantischen Ähnlichkeit von Beschriftungen das Maß von Wu und Palmer (1994) verwendet. In (Weidlich u. a. 2013) werden neben den Beschriftungen von Aktivitäten insbesondere weitere textuelle Beschreibungen in den Modellen berücksichtigt. Zum Auffinden von Matches werden Sprachmodelle (Lv und Zhai 2009) eingesetzt, die für das Information Retrieval verwendet werden. Die Triple-S Ansätze benötigen im Gegensatz dazu keine zusätzlichen Annotationen oder Beschreibungen.

Auch in (Baumann u. a. 2014; Baumann u. a. 2015b) werden zusätzliche Informationen wie etwa beteiligte Rollen oder Datenobjekte für das Auffinden von Matches hinzugezogen. Darüber hinaus wird in (Fengel und Reinking 2012; Fengel 2014) ein Ansatz beschrieben, um ähnliche Aktivitäten in Modellen unterschiedlicher Modellierungssprachen automatisch zu identifizieren. Dazu werden Techniken zur natürlichen Sprachverarbeitung sowie eine Überführung von Modellen in ein Ontologieformat eingesetzt. Für das Matching von Aktivitäten werden nur die Beschriftungen miteinander verglichen. Dazu kommen mehrere Techniken zur Verarbeitung natürlicher Sprache zum Einsatz. U. a. werden Stoppwörter entfernt, eine Wortstammreduktion durchgeführt und Synonyme mit Hilfe eines Thesaurus ermittelt.

Klinkmüller u. a. (2013) beschreiben einen weiteren Ansatz, der auf die Berücksichtigung der Modellstruktur verzichtet und lediglich auf die Beschriftungen von Aktivitäten zur Bestimmung von Matches zurückgreift. Dabei nutzen sie eine sogenannte Bag-of-Words Technik, die Beschriftungen in ihre Wörter zerlegt und basierend auf der Ähnlichkeit der Wörter bestimmt, ob es sich bei den Aktivitäten um ein Match handelt. In einer Variante davon werden zudem Wörter mit hoher Spezifität entfernt. Für die eigentliche Berechnung der Wortähnlichkeiten werden verschiedene Techniken eingesetzt, u. a. die Levenshtein-Distanz, das Ähnlichkeitsmaß von Lin (1998) und eine Wortstammreduktion. Demzufolge ist dieser Matching-Ansatz sehr ähnlich zu den Triple-S Verfahren, die ebenfalls auf die Levenshtein-Distanz und semantische Ähnlichkeitsberechnungen zwischen Wörtern in Beschriftungen zurückgreifen. Allerdings wird zusätzlich die Position von Aktivitäten in einem Modell berücksichtigt. Darüber hinaus nutzt ein von Klinkmüller u. a. (2014) beschriebener Ansatz neben einer automatischen Berechnung von Matches menschliche Eingaben zur Korrektheit dieser Matches, um die Qualität durch ein lernendes System zu verbessern. Von Rodriguez u. a. (2016) wurden zusätzlich Experimente durchgeführt, in denen die Qualität automatisierter Matching-Verfahren mit den Ergebnissen von Prozessmodellierungsexperten und mit von Crowd-Workern bestimmten Matches verglichen wird.

Weitere Ansätze wurden schließlich in den beiden Process Model Matching Wettbewerben veröffentlicht (Cayoglu u. a. 2014; Antunes u. a. 2015).[5] Im Folgenden werden aus diesen Veröffentlichungen die Verfahren beschrieben, die zuvor noch nicht erwähnt wurden. Dabei existiert eine Gruppe von Verfahren, die als RefMod-Mine Ansätze bezeichnet werden. Zwei stammen von Thaler et al. bzw. Hake et al. und sind in (Cayoglu u. a. 2014) beschrieben. Der Ansatz von Thaler et al. nutzt ein n-stufiges Clusterverfahren, bei dem im Gegensatz zu den üblichen paarweisen Vergleichen die Aktivitäten aller Modelle einer Sammlung gemeinsam überprüft werden. Zur Berechnung der Ähnlichkeit von Aktivitäten werden nur die Beschriftungen berücksichtigt, wobei letztlich die Wörter der Beschriftungen nach einer Wortstammreduktion verglichen werden. In (Antunes u. a. 2015) wird dieses Vorgehen ergänzt, um auch Synonyme in den Beschriftungen und organisationale Rollen zu berücksichtigen. Der Ansatz von Hake et al. verwendet ebenso die Ähnlichkeit von Wörtern in Aktivitätsbeschriftungen, allerdings werden bei diesem Verfahren die Modelle paarweise verglichen.

[5] Bei diesen Wettbewerben wurde u. a. auch die erste Triple-S Variante in der ursprünglichen Form verwendet.

Des Weiteren wird statt eines Clusteralgorithmus eine heuristische Gruppierung der Aktivitäten basierend auf Regeln und einem Ähnlichkeitsschwellenwert zur letztendlichen Berechnung von komplexen Matches eingesetzt.

Zusätzlich wurden in (Antunes u. a. 2015) drei weitere RefMod-Mine Verfahren veröffentlicht. Der erste Ansatz vergleicht nur die Beschriftungen von Aktivitäten durch ein dreistufiges Verfahren ohne die Struktur von Modellen zu berücksichtigen. In der ersten Stufe werden Aktivitätspaare als Match ermittelt, sofern die Beschriftungen exakt übereinstimmen oder wenn die eine Beschriftung in der anderen enthalten ist. In der zweiten Stufe werden die zu überprüfenden Beschriftungen in ihre Token zerlegt und deren Übereinstimmung zur Bestimmung von Matches genutzt. In der dritten Stufe werden die Beschriftungen schließlich über ein Vektormodell verglichen. Beim zweiten Ansatz wird per Part-of-Speech Tagging bestimmt, welche Teile einer Beschriftung Substantiven, Verben oder Adjektiven entsprechen, sodass diese für die Match-Berechnung miteinander auf Entsprechungen untersucht werden können. Solche Entsprechungen ergeben sich, wenn die Wörter in den Beschriftungsteilen identisch sind oder es sich um Synonyme, Homonyme oder etymologisch verwandte Worte handelt. Der dritte Ansatz basiert auf einem überwachten Maschinenlernverfahren, bei dem Gewichte gelernt werden, die die Ähnlichkeit von Wörtern in Aktivitätsbeschriftungen darstellen. Diese Gewichte werden mit Hilfe eines Goldstandards korrekter Matches gelernt und können anschließend für die Erkennung neuer Matches genutzt werden. Die Triple-S Ansätze verlangen hingegen keinen existierenden Goldstandard und berücksichtigen zudem die Struktur von Prozessmodellen, sind jedoch bezüglich der Ähnlichkeitsberechnung von Beschriftungen verwandt zu den RefMod-Mine Ansätzen. So wurde z. B. auch versucht Wortähnlichkeiten aus den Modellbeschriftungen mit Hilfe von WORD2VEC (Mikolov u. a. 2013b) zu bestimmen.

In (Antunes u. a. 2015) wurden darüber hinaus noch sechs weitere neue Verfahren beschrieben. Der erste Ansatz, *AgreementMakerLight*, wurde ursprünglich für das Matching von Ontologien entwickelt und für den Prozessmodell-Matching Wettbewerb adaptiert. Zunächst werden dabei aus den Prozessmodellen Ontologien generiert, wobei letztlich die Beschriftungen von Aktivitäten auf lexikalischer Ebene verglichen werden. Die Struktur eines Prozessmodells oder das Ablaufverhalten werden bei diesem Verfahren nicht berücksichtigt. Ein zweiter Ansatz, *KnowMa-Proc*, nutzt Techniken des Information Retrieval. In KnowMa-Proc werden nicht nur die Beschriftungen von Aktivitäten für das Matching herangezogen, sondern auch die Beschriftungen der direkten Vorgänger- und Nachfolgeraktivitäten. Aus diesen Beschriftun-

gen wird ein Index für ein Prozessmodell erstellt, der anschließend als Anfrage mit dem Index eines anderen Modells verglichen wird, um Matches zu identifizieren. Es wird nicht weiter ausgeführt, wie die Identifikation von Matches genau abläuft.

Die Ansätze drei und vier sind eng miteinander verwandt und nutzen Techniken der natürlichen Sprachverarbeitung. Unter anderem werden dabei Stoppwörter entfernt und die Ähnlichkeit von Beschriftungen über eine String-Editier Distanz und das Ähnlichkeitsmaß von (Lin 1998) berechnet. Somit wird auch bei diesen Ansätzen die Struktur von Modellen nicht berücksichtigt. Ansatz fünf, *OPBOT*, basiert auf den vorhergehenden Arbeiten von (Klinkmüller u. a. 2013; Klinkmüller u. a. 2014) und ergänzt diese um eine Betrachtung der Reihenfolgenbeziehungen von Aktivitäten durch die Berücksichtigung der relativen Distanz von Aktivitäten zum Startknoten eines Modells. In dieser Hinsicht ist OPBOT sehr ähnlich zu den Triple-S Ansätzen, die ebenfalls die relative Position von Aktivitäten in den Modellen berücksichtigen. Zusätzlich werden auch Rollenbezeichnungen zur Identifikation von Matches verwendet sowie ein Greedy-Algorithmus zur Optimierung eines Mappings eingesetzt. Der sechste Ansatz, *pPalm-DS*, berücksichtigt wiederum nur die Beschriftungen von Aktivitäten. Dazu wird zu jedem Wort ein Vektor erstellt, der zu diesem Wort häufig vorkommende andere Wörter enthält. Auf Basis dieser Vektoren wird daraufhin die Ähnlichkeit von Beschriftungen ermittelt.

Schließlich erweitern Niesen u. a. (2016) ihren Ansatz aus dem Process Model Matching Contest 2015. Durch ein phasenweises Vorgehen werden dabei Matches anhand der Beschriftungen berechnet. Weitere Dimensionen werden nicht berücksichtigt. In der ersten Phase werden Aktivitäten mit identischen Beschriftungen oder wenn eine Beschriftung ein Teil der anderen ist als Match klassifiziert. Phase zwei bestimmt Matches durch einen Vergleich der Lemmata von Beschriftungen. In Phase drei werden Beschriftungen schließlich in einen Vektorraum abgebildet und durch die Kosinus-Ähnlichkeit verglichen.

Zudem stellen van der Aa u. a. (2017) sechs Metriken vor, mit denen Informationen aus Traces für das Matching von Prozessmodellen eingesetzt werden können. Diese Metriken berücksichtigen z. B. die Häufigkeit und Dauer von Aktivitäten in den Traces zweier Prozessmodelle. Des Weiteren erörtern Meilicke u. a. (2017) den Einsatz eines Ensemble Matching Ansatzes, bei dem nicht nur ein sondern mehrere Matching-Verfahren eingesetzt werden. Dies stellt eine Lösungsmöglichkeit dar, um nicht ein spezifisches Matching-Verfahren für einen Datensatz einsetzen zu müssen, sondern

Matches durch mehrere Ansätze berechnen zu lassen. Aus der Gesamtheit aller berechneten Matches werden schließlich diejenigen ausgewählt, die mit hoher Wahrscheinlichkeit korrekt sind. Schließlich beschreiben Klinkmüller und Weber (2017) ein Vorgehen, um die beste Parameterkombination von Matching-Verfahren automatisiert zu bestimmen.

4.3 Die Familie der Triple-S Matching-Ansätze

Die in diesem Teilkapitel beschriebenen Ansätze zum Prozessmodell-Matching basieren auf dem in Algorithmus 4.1 dargestellten Ablauf. Dabei werden die Transitionen zweier beschrifteter Workflow-Netze (siehe Definition 2.7) auf ihre Ähnlichkeit hin überprüft. Generell nutzen beide beschriebenen Triple-S Ansätze Parameter und Gewichtungsfaktoren, die in den Zeilen 2–5 festgelegt werden. Der Parameter stellt einen Schwellenwert dar, ab dem zwei Transitionen als ausreichend ähnlich gelten, sodass sie ein Match ergeben. Anschließend werden drei Gewichtungsfaktoren für die drei verwendeten Dimensionen der Ähnlichkeitsberechnung festgelegt. Zu den berücksichtigten Dimensionen gehören die syntaktische und semantische Ähnlichkeit der Beschriftungen der Transitionen sowie eine strukturelle Ähnlichkeitsbewertung. Wie diese letztlich ausgestaltet sind, wird in den folgenden Teilkapiteln für jeden Ansatz separat beschrieben.

In den Zeilen 6–12 wird die Menge an Matches Λ und damit ein Matching entsprechend Definition 4.1 berechnet. Dazu wird zu jeder Transition des einen Modells die Ähnlichkeit zu jeder Transition des anderen Modells bestimmt. Die If-Bedingung in Zeile 13 entscheidet, ob zwei Transitionen ein Match ergeben. Sobald die gewichtete Summe der drei Ähnlichkeitswerte gleich groß oder höher als der Schwellenwert ist, werden die Transitionen als ein Match zur Menge aller Matches Λ hinzugefügt. Zwar werden jeweils nur zwei Transitionen miteinander verglichen, es können sich jedoch komplexe $1 : n$ oder $m : n$ Matches ergeben. $1 : n$ Matches entstehen, wenn zu einer Transition aus T_1 die Ähnlichkeitswerte zu mehreren Transitionen aus T_2 gleich groß oder größer als der festgelegte Schwellenwert sind. Zudem treten $m : n$ Matches auf, wenn zu mehreren Transitionen aus T_1 mehrere Transitionen aus T_2 einen Ähnlichkeitswert aufweisen, der gleich groß oder größer als der Schwellenwert ist.

Algorithmus 4.1 : Grundablauf der Triple-S Algorithmen

input : Two labelled Workflow Nets BWN_1 and BWN_2
output : A set of matches Λ between the power sets of the transitions of BWN_1 and BWN_2

```
1  Λ = ∅;
   /* Set threshold and weights. */
2  θ = threshold;
3  ω1 = synWeight;
4  ω2 = semWeight;
5  ω3 = strucWeight;
   /* Calculate similarity values for each dimension */
6  for each (t1 ∈ T1) do
7      for each (t2 ∈ T2) do
8          sim_syn = CalculateSyntacticalSimilarity(ℓ(t1), ℓ(t2));
9          sim_sem = CalculateSemanticSimilarity(ℓ(t1), ℓ(t2));
10         sim_struc = CalculateStructuralSimilarity(t1, t2);
11     end
12 end
   /* Determine final matches */
13 if (ω1 · sim_syn + ω2 · sim_sem + ω3 · sim_struc ≥ θ) then
14     Add match λ = (t1, t2) to Λ;
15 end
16 return Λ;
```

Formal betrachtet ergibt sich die Berechnung der Gesamtähnlichkeit zweier Transitionen $Sim_{ges}(t_1, t_2)$ als gewichtete Summe der Ähnlichkeitswerte der drei berücksichtigten Dimensionen:

$$Sim_{ges}(t_1, t_2) = \omega_1 \cdot sim_{syn}(\ell(t_1), \ell(t_2)) + \omega_2 \cdot sim_{sem}(\ell(t_1), \ell(t_2)) + \omega_3 \cdot sim_{str}(t_1, t_2). \tag{4.1}$$

Sim_{syn}, sim_{sem} und sim_{str} bilden dabei jeweils die Ähnlichkeitswerte auf das Intervall $[0, 1]$ ab. Die Gewichtungsfaktoren für die Ähnlichkeitswerte der drei Dimensionen (ω_1, ω_2 und ω_3) sind individuell bestimmbar. Um auch die Gesamtähnlichkeit auf das üblicherweise verwendete Intervall $[0, 1]$ zu begrenzen, müssen die beiden folgenden Bedingungen gelten:

- Die Wertebereiche der Faktoren müssen zwischen 0 und 1 liegen ($\omega_i \in [0, 1]$).
- Die Summe der Gewichtungsfaktoren muss 1 ergeben ($\sum_i \omega_i = 1$).

Schließlich wird ein Schwellwert θ verwendet, um zu bestimmen, ob die Gesamtähnlichkeit zweier Transitionen ausreichend hoch ist. Sollte $Sim_{ges}(t_1, t_2) \geq \theta$ sein, so

ergeben die Transitionen t_1 und t_2 ein Match. θ muss dabei ebenfalls im Wertebereich $[0, 1]$ liegen.

Die grundlegenden Ideen der Triple-S Verfahren sind demnach, dass ähnliche Aktivitäten eine ähnliche Wortwahl aufweisen und dass sie aus Sicht der Graphstruktur-Dimension eine ähnliche Position in dem jeweiligen Prozessmodell einnehmen. In den folgenden Teilkapiteln wird für die beiden entwickelten Triple-S Varianten beschrieben, wie die Funktionen in den Zeilen 8–10 umgesetzt werden, d. h., wie die Ähnlichkeit von Aktivitäten für jede Dimension quantifiziert werden kann. Zudem werden eventuelle Änderungen an der Bedingung in Zeile 13 erläutert, da beispielsweise die erste Triple-S Variante für die strukturelle Dimension zwei Aspekte berücksichtigt.

4.3.1 Triple-S

Der Triple-S Ansatz für das Matching von Prozessmodellen wurde im Rahmen zweier Wettbewerbe eingesetzt (Cayoglu u. a. 2014; Antunes u. a. 2015) und wird im Folgenden leicht adaptiert beschrieben. Dabei werden wie zuvor erläutert Transitionen hinsichtlich dreier unabhängiger Dimensionen auf ihre Ähnlichkeit hin bewertet. Bei diesen drei Dimensionen handelt es sich um die Syntax und Semantik der Transitionsbeschriftungen (syntaktische und semantische Dimension) sowie die Position der Transitionen in den Prozessmodellen (generelle Graphstruktur Dimension). Für die letzte Dimension werden zwei Aspekte berücksichtigt, sodass schließlich vier Ähnlichkeitswerte kombiniert werden, um einen finalen Ähnlichkeitswert zu berechnen. Diesem Matching-Ansatz liegt die Idee zugrunde, ein Verfahren zu entwerfen, das auf etablierten Techniken aufbaut und für verschiedene Modellierungssprachen verwendbar ist. Der entwickelte Ansatz kann als Basis für einen Vergleich komplexerer Verfahren angesehen werden, da letztlich in dem Triple-S Ansatz bekannte Techniken für die einzelnen berücksichtigten Dimensionen kombiniert werden.

Hinsichtlich der *syntaktischen Dimension* (sim_{syn}) werden zunächst drei Vorverarbeitungsschritte durchgeführt, indem die Beschriftungen in ihre einzelnen Token (Wörter) zerlegt werden, alle Wörter klein geschrieben und anschließend Stoppwörter entfernt werden. Die eigentliche Berechnung des Ähnlichkeitswerts der syntaktischen Dimension erfolgt daraufhin durch die Berechnung der auf das Interval $[0, 1]$ normierten Levenshtein-Distanz (Levenshtein 1966) zwischen jeder Kombination von Wörtern der betrachteten Beschriftungen. Das heißt, die Beschriftungen werden zunächst in einzelne Wörter zerlegt, um anschließend für jede Kombination von Wörtern der

Beschriftungen – und nicht für die vollständigen Beschriftungen – die Levenshtein-Distanz zu berechnen.

Dies wird durchgeführt, um Beschriftungen mit den gleichen oder verwandten Wörtern, aber abgeänderter Wortreihenfolge, erkennen zu können. So ergibt sich für die beiden Beschriftungen „determine test result“ und „test result determination“ bei einem Vergleich der einzelnen Wörter eine geringere Levenshtein-Distanz als für für die vollständige Beschriftung (im Beispiel 5 anstatt 18). Da die beiden Beschriftungen bis auf die unterschiedliche Wortreihenfolge und die Verwendung des Substantivs zu „determine“ gleich sind, wird dies als angemessener erachtet.

Sei $Lev(w_1, w_2)$ die Levenshtein-Distanz zwischen zwei Wörtern w_1 und w_2 und sei des Weiteren $len(w)$ die Anzahl an Zeichen eines Wortes, so berechnet sich die auf das Intervall $[0, 1]$ normierte Levenshtein-Distanz $Lev_{norm}(w_1, w_2)$ folgendermaßen:

$$Lev_{norm}(w_1, w_2) = \frac{Lev(w_1, w_2)}{max(len(w_1), len(w_2))}. \tag{4.2}$$

Der letztendliche Distanzwert zweier Beschriftungen $l(t_1)$ und $l(t_2)$ ergibt sich durch die Anwendung einer Greedy-Strategie, indem schrittweise jeweils eine Wortkombination aus bislang nicht berücksichtigten Wörtern mit der geringsten Distanz ausgewählt wird. Beschriftungen, die zwar die gleichen Wörter enthalten und sich nur deren Reihenfolge unterscheidet, würden ansonsten eine hohe Levenshtein-Distanz aufweisen, wie oben exemplarisch ausgeführt wurde.

Die Distanzen dieser Wortkombinationen werden daraufhin zu einer Gesamtdistanz $Lev_{ges}(l(t_1), l(t_2))$ addiert. Sollte eine der beiden verglichenen Beschriftungen mehr Wörter enthalten als die andere, wird für jedes zusätzliche Wort die Gesamtdistanz um 1 erhöht. Es wird also die größtmögliche Distanz zur Gesamtsumme hinzugefügt. Dies entspricht damit der Distanzberechnung Lev_{norm} zwischen einem Wort und einer leeren Zeichenkette. Die angewendete Greedy-Strategie kann dabei in einem lokalen Optimum resultieren, wird allerdings aufgrund der Problematik ausgewählt, dass anderenfalls alle möglichen Kombinationen von Wörtern getestet werden müssen, um ein globales Optimum zu ermitteln.

Aus der Gesamtdistanz der Wortkombinationen wird die durchschnittliche Distanz für die größere Anzahl an Wörtern in einer der beiden Beschriftungen berechnet, was

den finalen Distanzwert zweier Beschriftungen in Bezug auf die syntaktische Dimension darstellt. Sei $len(\ell(t))$ die Anzahl an Wörtern der Transitionsbeschriftung von Transition t nach den Vorverarbeitungsschritten, so ergibt sich die durchschnittliche Distanz Lev_{avg} zu:

$$Lev_{avg}(\ell(t_1), \ell(t_2)) = \frac{Lev_{ges}(\ell(t_1), \ell(t_2))}{max(len(\ell(t_1)), len(\ell(t_2)))}. \tag{4.3}$$

Der Ähnlichkeitswert aus syntaktischer Dimension ergibt sich dann letztendlich, indem von dem maximalen Ähnlichkeitswert von 1 der durchschnittliche Distanzwert Lev_{avg} abgezogen wird. Der finale Ähnlichkeitswert für die syntaktische Dimension sim_{syn} berechnet sich daher folgendermaßen:

$$sim_{syn}(\ell(t_1), \ell(t_2)) = 1 - Lev_{avg}(\ell(t_1), \ell(t_2)). \tag{4.4}$$

Wird etwa die Beschriftung der Transition „Ware verpacken“ aus Modell a) mit der Beschriftung der Transition „Ware versenden“ aus Modell b) in Abbildung 4.1 hinsichtlich der syntaktischen Dimension verglichen, so werden die Beschriftungen zunächst in ihre einzelnen Wörter zerlegt und klein geschrieben. Anschließend wird für jede Kombination von Wörtern der beiden Beschriftungen die normierte Levenshtein-Distanz Lev_{norm} berechnet. Im Beispiel sind diese 0 für die Kombination aus „ware“ und „ware“, $\frac{8}{9}$ für „ware“ und „versenden“ bzw. „verpacken“ und „ware“ sowie $\frac{4}{9}$ für „verpacken“ und „versenden“. Somit werden als Wortkombinationen „ware“ und „ware“ sowie „verpacken“ und „versenden“ durch die Greedy-Strategie ausgewählt, sodass sich eine Gesamtdistanz von $\frac{4}{9}$ ergibt. Die durchschnittliche Gesamtdistanz Lev_{avg} ist demnach $\frac{2}{9}$, da beide Beschriftungen zwei Wörter enthalten. Der Ähnlichkeitswert sim_{syn}(„Ware verpacken“, „Ware versenden“) beträgt schließlich $1 - \frac{2}{9} = \frac{7}{9}$.

Auch bezüglich der *semantischen Dimension* (sim_{sem}) werden zunächst die gleichen Vorverarbeitungsschritte wie für die syntaktische Dimension durchgeführt. Daraufhin wird die semantische Ähnlichkeit zwischen den Wörtern der Transitionsbeschriftungen nach dem Ansatz von Wu und Palmer (1994) berechnet (siehe auch Kapitel 3.2). Die einzelnen Wortähnlichkeiten liegen nach (Wu und Palmer 1994) jeweils im Intervall $[0, 1]$. Der letztendliche Ähnlichkeitswert ergibt sich ebenso analog zur syntaktischen Ähnlichkeit durch die Anwendung einer Greedy-Strategie. Dabei wird

in jedem Schritt die Wortkombination mit der höchsten Ähnlichkeit ausgewählt und zu einer Gesamtähnlichkeit Sim_{WuGes} der verglichenen Beschriftungen addiert. Sollte eine Beschriftung mehr Wörter enthalten als die andere, wird für jedes zusätzliche Wort weder etwas von der Gesamtähnlichkeit Sim_{WuGes} abgezogen noch addiert. Hier unterscheidet sich die Berechnung der semantischen Ähnlichkeit von der syntaktischen, bei der für jedes zusätzliche Wort die Distanz um 1 erhöht wird.

Der finale Ähnlichkeitswert der semantischen Dimension ergibt sich schließlich als die durchschnittliche Ähnlichkeit aller Wortkombinationen. Für die Gesamtähnlichkeit der Beschriftungen $Sim_{WuGes}(l(t_1), l(t_2))$ zweier Transitionen t_1 und t_2 ergibt sich der Ähnlichkeitswert der semantischen Dimension sim_{sem} als:

$$sim_{sem}(\ell(t_1), \ell(t_2)) = \frac{Sim_{WuGes}(\ell(t_1), \ell(t_2))}{max(len(l(t_1)), len(l(t_2)))}. \tag{4.5}$$

Durch die Verwendung dieser Dimension soll die bei der Modellierung auftretende Verwendung von Synonymen oder verwandten Wörtern berücksichtigt werden. Beispielsweise kann dadurch automatisiert erkannt werden, dass die Transition „Ware versenden“ aus Modell a) in Abbildung 4.1 ähnlich zu einer Transition „Ware verschicken“ ist, die zu dem Wort „versenden“ das Synonym „verschicken“ enthält.

Bezüglich der *strukturellen Dimension* werden zwei unterschiedliche Aspekte zur Berechnung der Ähnlichkeit von Transitionen betrachtet. (1) sim_{str1}: Das Verhältnis der ein- und ausgehenden Kanten sowie (2) sim_{str2}: die relative Position innerhalb der Modelle. Hinsichtlich des ersten Aspekts sind die Ähnlichkeitswerte beim Vergleich der Transitionen „Bestellung vorbereiten“ aus Modell a) und „Ware versenden“ aus Modell b) in Abbildung 4.1 1,0 für das Verhältnis eingehender und 0,5 für das Verhältnis ausgehender Kanten. Die Gesamtähnlichkeit beträgt demzufolge 0,75.

Formal beschrieben wird die Gesamtähnlichkeit sim_{str1} wie in Formel 4.6 angegeben berechnet, wobei $I_t = |\bullet\, t| = |\{n_i \mid (n_i, t) \in F\}|$ und $O_t = |t \,\bullet| = |\{n_i \mid (t, n_i) \in F\}|$ die Anzahl eingehender bzw. ausgehender Kanten einer Transition bezeichnen:

$$sim_{str1}(t_1, t_2) = 0,5 \cdot (\frac{min(I_{t_1}, I_{t_2})}{max(I_{t_1}, I_{t_2})} + \frac{min(O_{t_1}, O_{t_2})}{max(O_{t_1}, O_{t_2})}). \tag{4.6}$$

Hinsichtlich der relativen Position von Transitionen in einem Modell wird das Verhältnis der Länge des kürzesten Pfads von der Quelle i zu einer Transition t ($len(\rho_1)$)

und der Länge des kürzesten Pfads von t zur Senke o ($len(\rho_2)$) verwendet. Dieser relativen Positionsberechnung liegt die Idee zugrunde, dass Transitionen weniger ähnlich sind, wenn sie an unterschiedlichen „Enden“ eines Prozessmodells vorkommen, selbst wenn sie sehr ähnliche oder gleiche Beschriftungen aufweisen. Zudem werden die Längen von kürzesten Pfaden verglichen, um die Problematik von Zyklen in Prozessmodellen zu lösen. Beispielsweise führt ein Zyklus zwischen der Quelle i und einer Transition t zu einem theoretisch unendlich langen Pfad, je nachdem wie oft der Zyklus durchlaufen wird. Durch die Verwendung von kürzesten Pfaden wird die Länge entsprechend begrenzt. Für eine Transition t_1 aus einem Prozessmodell m_1 wird daher die relative Position entsprechend folgender Formel berechnet:

$$RelPos(t_1, m_1) = \frac{len(\rho_1)}{len(\rho_1) + len(\rho_2)}. \tag{4.7}$$

Zum Vergleich von zwei Transitionen t_1 aus Modell m_1 und t_2 aus Modell m_2 wird schließlich die jeweilige relative Position verglichen. Formal beschrieben berechnet sich sim_{str2} somit folgendermaßen:

$$sim_{str2}(t_1, t_2) = \frac{min(RelPos(t_1, m_1), RelPos(t_2, m_2))}{max(RelPos(t_1, m_1), RelPos(t_2, m_2))}. \tag{4.8}$$

Die Werte von sim_{str2} liegen damit ebenfalls im Intervall $[0, 1]$. Ein niedriger Wert von $RelPos$ bedeutet, dass die betrachtete Transition eine kurze Distanz zur Quelle i hat, während ein hoher Wert das Gegenteil ausdrückt. Somit drückt ein niedriger Wert aus, dass die Transition eher am „Anfang“ eines Prozessmodells liegt und ein hoher Wert repräsentiert eine Transition, die eher am „Ende“ eines Prozessmodells liegt.

Für die zuvor betrachteten Transitionen t_1 „Bestellung vorbereiten“ und t_2 „Ware versenden“ hat $len(\sigma_1)$ den Wert 3 bzw. 5 und $len(\sigma_2)$ den Wert 7 bzw. 3. Daraus ergeben sich die relativen Positionswerte $RelPos(t_1) = \frac{3}{3+7} = 0,3$ und $RelPos(t_2) = \frac{5}{5+3} = 0,625$. Als struktureller Ähnlichkeitswert dieses Aspekts ergibt sich schließlich $sim_{str2}(t_1, t_2) = \frac{min(0,3,0,625)}{max(0,3,0,625)} = \frac{0,3}{0,625} = 0,48$.

Die allgemeine Formel zur Bestimmung, ob zwei Transitionen ausreichend ähnlich sind, um ein Match zu ergeben (siehe Formel 4.1), wird folglich für die erste Triple-S Variante geringfügig abgeändert. Statt eines Wertes für die strukturelle Dimension werden zwei Werte verwendet. Daher ergibt sich die angepasste Bedingung zu:

$$sim_{ges}(t_1, t_2) = \omega_1 \cdot sim_{syn}(\ell(t_1), \ell(t_2)) + \omega_2 \cdot sim_{sem}(\ell(t_1), \ell(t_2)) + \omega_3 \cdot sim_{str1}(t_1, t_2) + \omega_4 \cdot sim_{str2}(t_1, t_2). \tag{4.9}$$

ω_4 muss dabei entsprechend auch im Intervall $[0, 1]$ liegen und die Summe der Gewichtungsfaktoren $\omega_i, i = 1, ..., 4$ muss 1 ergeben.

4.3.2 Triple-S2

Basierend auf den Evaluationsergebnissen in den beiden Process Model Matching Wettbewerben (Cayoglu u. a. 2014; Antunes u. a. 2015) wurde der Triple-S Ansatz adaptiert, um die Qualität der Matching-Ergebnisse zu verbessern. Diese Adaptionen sind im Folgenden beschrieben und betreffen vier Aspekte:

(i) Bei der Berechnung der syntaktischen Ähnlichkeit wird die Levenshtein-Distanz nicht für die Wörter von Beschriftungen berechnet, sondern für die Wortstämme. Der Wortstamm wird nach dem Porter-Algorithmus (Porter 1980) berechnet und ermöglicht die Erkennung unterschiedlicher Wortformen. So ist sowohl für „determine“ als auch „determination“ der Wortstamm jeweils „determin“, sodass die Levenshtein-Distanz nun 0 beträgt und nicht 5 wie beim Vergleich der ursprünglichen Worte. Dadurch kann beispielsweise erkannt werden, dass die beiden Beschriftungen „determine test result“ und „test result determination“ die gleiche Aktivität bezeichnen.

(ii) Für die semantische Dimension wird WORD2VEC (Mikolov u. a. 2013b) als Ähnlichkeitsmaß für Wörter verwendet, da einige der in den Modellen verwendeten Begriffe nicht in WORDNET (Fellbaum 1998) enthalten sind (z. B. citizen center für Bürgeramt) und die semantische Ähnlichkeit zu anderen Wörtern bei dem Triple-S Ansatz damit 0 beträgt. Dies ist generell der Fall für Wörter, die nicht in WORDNET oder anderen Korpora enthalten sind und trifft insbesondere auch für unternehmensspezifische Begriffe zu. Diese Begriffe müssten manuell zu einem Korpus hinzugefügt werden, während WORD2VEC den Vorteil bietet, Wortähnlichkeiten aus existierenden Dokumenten, die diese Begriffe enthalten, bestimmen zu können.

(iii) Bei der Berechnung der strukturellen Ähnlichkeit erwiesen sich die Werte für das Kantenverhältnis als wenig unterscheidungskräftig, sodass ein Vergleich der strukturellen Dimension nur die relative Position von Transitionen berücksichtigt.

(iv) Sowohl die Ähnlichkeitsbestimmung auf syntaktischer als auch auf semantischer Ebene zielen auf die Beschriftungen von Transitionen ab. In dem Triple-S Verfahren wurden die jeweiligen Ähnlichkeitswerte gewichtet addiert, es erscheint jedoch sinnvoller diese separat zu betrachten.

Hinsichtlich des vierten Aspekts zeigt Abbildung 4.3 ein Beispiel für die Sinnhaftigkeit der Trennung von syntaktischer und semantischer Ebene. Enthalten z. B. Beschriftungen die gleichen Wörter, so beträgt sowohl die syntaktische als auch die semantische Ähnlichkeit 1, d. h., die semantische Ähnlichkeit wirkt wie ein höheres Gewicht der syntaktischen Ähnlichkeit. In Abbildung 4.3 ist dies etwa der Fall, wenn die beiden Transitionen „Send package“ miteinander verglichen werden. Werden hingegen Transitionen verglichen, deren Beschriftungen Synonyme enthalten wie beispielsweise in Abbildung 4.3 bei „Send package“ und „Transport package“, so wirkt die syntaktische Ähnlichkeit wie ein negatives Gewicht für die semantische Ähnlichkeit. Angenommen, die semantische Wortähnlichkeit zwischen „Send“ und „Transport“ wäre 1, dann wäre auch die semantische Ähnlichkeit der vollständigen Beschriftungen 1. Die syntaktische Ähnlichkeit der Beschriftungen beträgt jedoch nur 0,55. Somit bewirkt die syntaktische Ähnlichkeit eine Reduktion der Gesamtähnlichkeit, obwohl die gleiche Aktivität durch beide Transitionen repräsentiert wird.

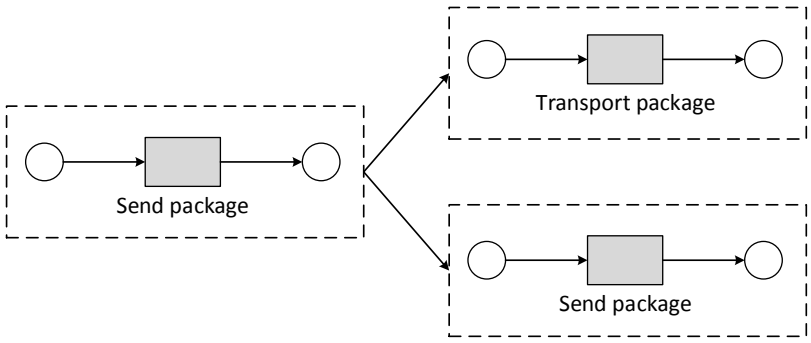


Abb. 4.3: Beispiel für die Aufteilung von syntaktischem und semantischem Vergleich von Beschriftungen

Algorithmus 4.2 zeigt den Ablauf des Triple-S2 Verfahrens. In den Zeilen 2–7 werden dabei die Gewichte und Schwellenwerte definiert. In den beiden *for each* Schleifen werden anschließend wiederum alle Kombinationen von Transitionspaaren miteinander verglichen, ob sie ein Match darstellen. Zunächst werden dazu in den Zeilen 10 und 11 die syntaktische[6] und strukturelle Ähnlichkeit der beiden untersuchten Transitionen berechnet. Sollte die gewichtete Summe gleich groß oder größer sein als der Schwellenwert θ_1, so werden sie als Match klassifiziert (Zeilen 12–14). Sollte dies nicht der Fall sein, wird die semantische Ähnlichkeit der beiden Transitionen berechnet (Zeile 16). Falls die gewichtete Summe der semantischen und strukturellen Ähnlichkeit gleich groß oder größer wie der Schwellenwert θ_2 ist, so bilden die beiden Transitionen ein Match (Zeilen 17–19). Sollte auch dies nicht der Fall sein, stellen die beiden Transitionen kein Match dar. In Zeile 23 werden die berechneten Matches schließlich zurückgegeben.

Algorithmus 4.2 : Triple-S2 Algorithmus

input : Two labelled Workflow Nets BWN_1 and BWN_2
output : A set of matches Λ between the power sets of the transitions of BWN_1 and BWN_2

```
1  Λ = ∅;
2  θ1 = synThreshold;
3  θ2 = semThreshold;
4  ω1 = synWeight;
5  ω2 = strucWeightSyn;
6  ω3 = semWeight;
7  ω4 = strucWeightSem; for each (t1 ∈ T1) do
8      for each (t2 ∈ T2) do
           /* Decide on syntactic match */
9          sim_syn = CalculateSyntacticalSimilarity(ℓ(t1), ℓ(t2));
10         sim_struc = CalculateStructuralSimilarity(t1, t2);
11         if (ω1 · sim_syn + ω2 · sim_struc ≥ θ1) then
12             Add match λ = (t1, t2) to Λ;
13         end
           /* Decide on semantic match */
14         else
15             sim_sem = CalculateSemanticSimilarity(ℓ(t1), ℓ(t2));
16             if (ω3 · sim_sem + ω4 · sim_struc ≥ θ2) then
17                 Add match λ = (t1, t2) to Λ;
18             end
19         end
20     end
21 end
22 return Λ;
```

[6] Die Berechnung der Levenshtein-Distanz erfolgt bei Triple-S2 allerdings nicht für die Wörter von Beschriftungen, sondern für die Wortstämme wie zuvor beschrieben.

Die oben erläuterte Trennung von syntaktischer und semantischer Ebene wird in Algorithmus 4.2 durch die If-Bedingungen realisiert, wobei zunächst ein Vergleich auf syntaktischer Ebene und anschließend ein Vergleich auf semantischer Ebene durchgeführt wird. Durch die beiden Schwellenwerte θ_1 und θ_2 kann darüber hinaus der geforderte Ähnlichkeitswert zwischen zwei Transitionen für beide Vergleiche individuell festgelegt werden. Ein weiterer Unterschied zum Triple-S Ansatz ist das in der Funktion *CalculateSemanticSimilarity()* eingesetzte Verfahren. Im Gegensatz zu Triple-S verwendet Triple-S2 für die Bestimmung von semantischen Wortähnlichkeiten nicht den Ansatz aus (Wu und Palmer 1994), sondern greift auf WORD2VEC zurück (Mikolov u. a. 2013b). Die Berechnung erfolgt allerdings entsprechend Formel 4.5. Der syntaktische Vergleich in der Funktion *CalculateSyntacticalSimilarity()* wird hingegen unverändert aus dem Triple-S Verfahren übernommen, die Berechnung entspricht daher Formel 4.4. Schließlich wird für den strukturellen Vergleich nur die relative Positionsberechung aus dem Triple-S Ansatz verwendet. Die Berechnung in der Funktion *CalculateStructuralSimilarity()* erfolgt daher entsprechend Formel 4.8.

4.4 Evaluation

In diesem Teilkapitel wird die Evaluation der zuvor erläuterten Prozessmodell-Matching Ansätze beschrieben. Dazu wird zunächst auf eine Implementierung dieser Ansätze sowie das verwendete Testsystem eingegangen, um anschließend die verwendeten Datensätze und Goldstandards zu erläutern. Schließlich werden die Ergebnisse der Evaluation beschrieben und diskutiert, wobei auch auf einen Vergleich mit zwei Verfahren aus dem Process Model Matching Contest 2015 (Antunes u. a. 2015) eingegangen wird.

4.4.1 Testsystem und -daten

Für die Realisierung der beiden Triple-S Ansätze[7] wurde Java 8 eingesetzt. Zusätzlich wurde die SEMILAR API (Rus u. a. 2013) zur Verarbeitung der Beschriftungen von Transitionen verwendet. Dabei wird zur Erzeugung von Wörtern bzw. Tokens aus

[7] Der Quellcode ist unter https://github.com/ASchoknecht/Triple-S verfügbar.

den Beschriftungen der Stanford Tokenizer[8] verwendet. Die Erkennung von Stoppwörtern wird ebenfalls durch die SEMILAR API[9] ermöglicht und basiert auf einer manuell erstellten englischsprachigen Liste. Es wurde keine existierende Liste übernommen, um für den Kontext des Matchings relevante Worte nicht auszuschließen. So werden teilweise Verben in Stoppwort-Listen ausgeschlossen, die allerdings beim Matching von Aktivitäten eine Rolle spielen. Die Beschriftungen von Transitionen drücken in Form von Verben insbesondere aus, welche Aktivität durchgeführt werden soll.

Für den Triple-S Matching-Ansatz wurde für die Berechnung des semantischen Ähnlichkeitswerts von Wörtern nach (Wu und Palmer 1994) eine entsprechende Funktion der SEMILAR API verwendet, die den Wert mit Hilfe von WordNet (Fellbaum 1998) berechnet. Für das Triple-S2 Verfahren wird zur Berechnung der Wortähnlichkeit die Word2Vec-Implementierung aus der Deeplearning4J-Bibliothek[10] eingesetzt. Die Wortähnlichkeiten basieren dabei auf einem von Google veröffentlichten Datensatz,[11] der insgesamt 3 Millionen Wörter enthält. Zur Bestimmung der relativen Position von Transitionen in den verwendeten beschrifteten Workflow-Netzen wird bei beiden Triple-S Verfahren der Dijkstra-Algorithmus (Dijkstra 1959) zur Berechnung kürzester Pfade genutzt.

Bei den weiteren verglichenen Verfahren handelt es sich zum einen um RefMod-Mine/NHM und zum anderen um RefMod-Mine/NLM, die beide in (Antunes u. a. 2015) beschrieben und über die RefMod-Miner Plattform[12] verfügbar sind. Über diese Plattform können Matches für Prozessmodelle berechnet werden, wobei für die beiden Verfahren außer einer Auswahl der verwendeten natürlichen Sprache in den Modellbeschriftungen keine Parametrierungsoptionen angeboten werden. Diese beiden Ansätze wurden ausgewählt, da sie bei dem Process Model Matching Contest 2015 (Antunes u. a. 2015) zu den besten Verfahren zählten.

Die Evaluationsdatensätze basieren auf den für den Process Model Matching Contest 2015 (Antunes u. a. 2015) erstellten Modelldatensätzen. Beide Datensätze enthal-

[8] http://nlp.stanford.edu/software/tokenizer.shtml
[9] http://deeptutor2.memphis.edu/Semilar-Web
[10] https://deeplearning4j.org
[11] Weitere Informationen zu dem Datensatz sind unter https://code.google.com/archive/p/word2vec verfügbar. Ein Download des Datensatzes ist unter https://s3.amazonaws.com/dl4j-distribution/GoogleNews-vectors-negative300.bin.gz möglich.
[12] http://refmod-miner.dfki.de

ten jeweils neun Modelle, woraus sich 36 zu vergleichende Modellpaare ergeben. Der erste Datensatz beinhaltet die Bewerbungsprozesse für Masterstudiengänge an neun deutschen Universitäten (University Admission Modelle UA). Die ursprünglich als BPMN-Diagramme vorliegenden Modelle wurden manuell vom Autor dieser Arbeit in beschriftete Workflow-Netze transformiert, um sie für die Triple-S Implementierungen verwenden zu können. Der zweite Datensatz enthält Geburtsregistrierungsprozesse aus Deutschland, Südafrika, Russland und den Niederlanden und besteht ebenfalls aus neun Modellen (Birth Registration Modelle BR). Eine Übersicht der Anzahl an Stellen, Transitionen, Knoten und Kanten für die beiden Datensätze ist in Tabelle 4.2 dargestellt.

Tab. 4.2: Minimale, maximale und durchschnittliche Anzahl an Stellen, Transitionen, Knoten und Kanten der Matching Evaluationsdatensätze

		Geburtsmodelle BR	Bewerbungsmodelle UA
Stellen	Min	12	14
	Max	35	49
	∅	26	30
Transitionen	Min	10	13
	Max	28	46
	∅	20	28
Knoten	Min	22	27
	Max	63	95
	∅	46	57
Kanten	Min	24	29
	Max	72	110
	∅	53	66

Für die Evaluierung wurden für jeden Datensatz jeweils fünf Goldstandards erstellt, um die Menge an korrekten Matches festzulegen.[13] Jeweils ein Goldstandard beider Datensätze entspricht dem in dem Process Model Matching Contest 2015 eingesetzten Goldstandard. Die vier anderen Standards wurden jeweils von Prozessmodellierungsexperten erstellt. Zur Erstellung der Standards wurden den Experten die Modelldatensätze sowie eine Anleitung übergeben. In dieser Anleitung wurden die zu erledigende Aufgabe und eine Definition von Matches beschrieben. Zudem ent-

[13] Die Modelle, Goldstandards und die berechneten Matches sowie Kennzahlen finden sich auch unter http://butler.aifb.kit.edu/asc/TripleS/MatchingEvaluation.zip.

hielt die Anleitung einige Beispiele möglicher Matches und es wurde die Möglichkeit für Rückfragen gegeben. Aus den letztlich fünf erhaltenen Goldstandards wurde anschließend ein Standard erstellt, in dem alle eindeutigen Matches sowie ihre Häufigkeit aufgelistet sind (Goldstandard Gesamt). Da sich bei der Untersuchung des Gesamtstandards ergab, dass viele Matches lediglich in einem oder zwei Goldstandards vorkommen, wurde noch ein weiterer Goldstand erstellt, in dem nur die Matches enthalten sind, die in mindestens drei Standards genannt wurden (Goldstandard Mehrheit). Dadurch soll verglichen werden, wie gut die automatisierte Match-Berechnung für die von der Mehrheit der Experten identifizierten Matches abschneidet, was als Grundkonsens aufgefasst wird.

Eine Übersicht der Anzahl an Matches pro Goldstandard ist in Tabelle 4.3 dargestellt. Neben der absoluten Anzahl sind die durchschnittliche Anzahl sowie die minimale und maximale Anzahl an Matches in einem Modellpaar aufgelistet. Bei einem Vergleich des Mehrheitsstandards mit dem Gesamtstandard fällt auf, dass bei dem University Admission Datensatz lediglich 47,8 % aller Matches in mindestens drei Standards vorkommen. Bei dem Birth Registration Datensatz sind es sogar nur 29,5 %. Diese Ergebnisse bestätigen die Erkenntnisse aus (Rodriguez u. a. 2016), da die vier von Experten erstellten Standards in deren Untersuchung ebenfalls eine geringe Übereinstimmung von 40 % bis 60 % aufweisen. Daher scheinen sich die in (Thaler u. a. 2014) publizierten Erläuterungen zur Unterschiedlichkeit von Goldstandards zu bestätigen.

Tab. 4.3: Charakteristika der Matching Goldstandards (GS)

University Admission	**GS 1**	**GS 2**	**GS 3**	**GS 4**	**GS 5**	**Gesamt**	**Mehrheit**
# Matches	479	212	405	347	271	651	311
∅ # Matches	13,31	5,89	11,25	9,64	7,53	18,08	8,64
Min # Matches	7	1	5	4	3	9	4
Max # Matches	30	22	25	29	22	38	24
Birth Registration	**GS 1**	**GS 2**	**GS 3**	**GS 4**	**GS 5**	**Gesamt**	**Mehrheit**
# Matches	262	583	218	279	150	699	206
∅ # Matches	7,28	16,19	6,06	7,75	4,17	19,42	5,72
Min # Matches	2	5	0	1	0	8	0
Max # Matches	24	27	25	25	21	31	25

In dieser Arbeit wird ein pragmatischer Ansatz für die Evaluation der Matching-Verfahren gewählt, um mit der Uneindeutigkeit von Matches umzugehen. Das heißt, das

Ziel der Evaluation ist die Erkennung möglichst vieler in dem Gesamtstandard enthaltener Matches bei möglichst wenig erkannten nicht enthaltenen Matches. Demzufolge wird nicht die Forderung erhoben, dass die Matches der Goldstandards zueinander konsistent sein müssen.[14]

Konsistent ist in diesem Zusammenhang so zu verstehen, dass für ein Match eines Goldstandards zwar eine nachvollziehbare Argumentation existieren muss, aus der ersichtlich wird, dass die Transitionen die gleiche Aktivität in der realen Welt repräsentieren. Aus dem Vorkommen eines solchen Matches kann aber nicht gefolgert werden, dass dieses Match auch in den anderen Standards existieren muss. Ein Beispiel aus dem University Admission Datensatz ist die Einschätzung, ob ein mündliches Bewerbungsgespräch einem Auswahltest entspricht. Man kann argumentieren, dass sowohl das Gespräch als auch der Test der Auswahl von passenden Studierenden gilt und die Aktivitäten sich somit entsprechen, d. h. ein Match darstellen. Wenn allerdings ein besonderer Fokus auf die Art der Auswahl gelegt wird, so kann sich der Schluss ergeben, dass sich die beiden Aktivitäten *nicht* entsprechen, da ein Test eine andere Form darstellt als ein Auswahlgespräch. Daher wird in der folgenden Evaluation nicht davon ausgegangen, dass eine objektive Wahrheit bezüglich der Entsprechung von Aktivitäten existiert, sondern die Matches werden gleichwertig berücksichtigt und keines ausgeschlossen. Damit soll die Beschränkung auf die Perspektive eines bestimmten Standards vermieden werden.

Die Häufigkeit eines Matches in einem Gesamt- bzw. Mehrheitsstandard stellt eine Gewichtung dar, um die Relevanz einzelner Matches unterscheiden zu können. Dabei liegt die Annahme zugrunde, dass häufig genannte Matches wichtiger sind als selten vorkommende. Die automatisierten Verfahren werden dementsprechend je nach Relevanz der erkannten Matches unterschiedlich bewertet. Diesem Gedanken folgend soll mit Hilfe des Mehrheitsstandards zudem überprüft werden, wie gut die häufig genannten Matches von den automatisierten Verfahren erkannt werden. Ein Mehrheitsstandard stellt in gewissem Sinn eine Konsensmeinung der Experten dar, da darin nur die Matches enthalten sind, die von einer Mehrheit an Experten genannt wurden. Dadurch soll eine Aussage über die Qualität der automatisierten Matching-Verfahren über die eher eindeutigen Matches ermöglicht werden.

[14] Es wurde zudem sichergestellt, dass in einem Standard keine redundanten oder offensichtlich falschen Matches enthalten sind, was z. B. durch eine Vertauschung von Transitions-IDs bei der Beschreibung von Matches vorkommen kann.

4.4.2 Evaluationsergebnisse

Die Matching-Berechnung wurde mit einem Laptop mit Intel i7-4750HQ Prozessor, 8 GB RAM mit Windows 8.1 Enterprise Version durchgeführt. Die Berechnung von Precision- und Recall-Werten wurde mit Hilfe der auch während des Process Model Matching Contest 2015 verwendeten ALIGNMENT API[15] berechnet. F-Werte sowie die Durchschnittswerte und Standardabweichung wurden anschließend manuell ermittelt. Für den Gesamtstandard sowie den Mehrheitsstandard wurden die Kennzahlen probabilistische Precision, probabilistischer Recall und probabilistischer F-Wert verwendet (für weitere Erläuterungen der Kennzahlen siehe Kapitel 3.4). Dazu werden True Positives mit der Häufigkeit gewichtet, in denen sie in den Goldstandards vorkommen. False Positives werden mit dem mittleren Faktor 3 gewichtet, um ihren Einfluss auszugleichen, sodass weder Precision noch Recall bevorzugt werden. Ein niedriges Gewicht legt tendenziell den Fokus auf den Recall, da False Positives einen relativ geringen Einfluss auf den F-Wert haben. Ein hohes Gewicht führt hingegen dazu, dass False Positives einen relativ großen Einfluss auf den F-Wert haben.

Die Bestimmung der Parameterwerte für die Triple-S Verfahren wurde entsprechend der Matching Wettbewerbe durchgeführt. Wie bei den Wettbewerben wurde der University Admission Datensatz zum Training der Parameter verwendet. Die gleichen Werte wurden anschließend für das Matching des Birth Registration Datensatzes genutzt. Als Goldstandard für das Training wurde in der folgenden Evaluation der Gesamtstandard verwendet. Es wurde schließlich die Parameterkombination ausgewählt, die den Micro Average F-Wert maximiert. Der Micro Average wurde gewählt, da sich die Anzahl an korrekten Matches pro Modellpaar stark unterscheiden kann. So existieren beispielsweise für den Birth Registration Datensatz in den Goldstandards einige Modellpaare für die überhaupt keine Matches identifiziert wurden, bei anderen Modellpaaren wurden hingegen 27 korrekte Matches bestimmt.

Die Ergebnisse für den University Admission Datensatz sind in Tabelle 4.4 dargestellt. Dabei ist zu erkennen, dass der Triple-S2 Ansatz hinsichtlich aller Kennzahlen besser abschneidet als das Triple-S Verfahren, wenn der Gesamtstandard bzw. der Mehrheitsstandard betrachtet wird. Bei den einzelnen Standards ist dies mit wenigen Ausnahmen ebenfalls der Fall. So übertrifft das Triple-S Verfahren den Triple-S2 Ansatz lediglich in Bezug auf den Macro AVG Precision Wert für Goldstandard 2 und

[15] Weitere Informationen unter http://alignapi.gforge.inria.fr/tutorial

den Micro AVG Recall Wert für Goldstandard 5. Bei einem Vergleich der Precison- und Recall-Werte der einzelnen Goldstandards fällt auf, dass sich Goldstandard 2 relativ stark von den anderen vier Standards unterscheidet. Während bei Goldstandard 2 die Precision-Werte deutlich geringer sind als bei den anderen Standards, sind die Recall-Werte deutlich höher. Die zugrundeliegende Einschätzung von Matches dieses Experten unterscheidet sich somit von den anderen Experten.

Tab. 4.4: Ergebnisse der Matching-Verfahren für den University Admission Datensatz

			GS1	GS2	GS3	GS4	GS5	Gesamt[a]	Mehrheit[a]
Triple-S[b]	Precision	Micro AVG	0,52	0,36	0,50	0,42	0,42	**0,62**	0,58
		Macro AVG	0,62	0,32	0,60	0,53	0,50	**0,68**	0,61
		STD	0,33	0,34	0,33	0,31	0,35	**0,30**	0,30
	Recall	Micro AVG	0,33	0,51	0,37	0,37	0,47	**0,39**	0,50
		Macro AVG	0,31	0,31	0,35	0,34	0,39	**0,34**	0,40
		STD	0,29	0,36	0,31	0,31	0,34	**0,31**	0,32
	F-Wert	Micro AVG	0,40	0,43	0,43	0,39	0,45	**0,48**	0,54
		Macro AVG	0,33	0,24	0,34	0,31	0,33	**0,37**	0,40
		STD	0,24	0,26	0,25	0,21	0,26	**0,26**	0,27
Triple-S2[c]	Precision	Micro AVG	0,61	0,40	0,56	0,49	0,56	**0,70**	0,63
		Macro AVG	0,70	0,26	0,66	0,57	0,66	**0,77**	0,67
		STD	0,28	0,26	0,25	0,27	0,25	**0,20**	0,23
	Recall	Micro AVG	0,37	0,55	0,40	0,41	0,40	**0,43**	0,54
		Macro AVG	0,36	0,37	0,40	0,39	0,40	**0,38**	0,45
		STD	0,33	0,38	0,30	0,30	0,30	**0,30**	0,29
	F-Wert	Micro AVG	0,46	0,46	0,47	0,44	0,47	**0,53**	0,58
		Macro AVG	0,40	0,29	0,42	0,39	0,42	**0,44**	0,48
		STD	0,24	0,29	0,21	0,18	0,21	**0,23**	0,21
NLM[d]	Precision	Micro AVG	0,90	0,78	0,78	0,78	0,78	**0,91**	0,88
		Macro AVG	0,87	0,75	0,72	0,74	0,73	**0,84**	0,81
		STD	0,24	0,26	0,33	0,28	0,29	**0,23**	0,23
	Recall	Micro AVG	0,23	0,44	0,23	0,27	0,35	**0,28**	0,37
		Macro AVG	0,22	0,36	0,21	0,25	0,28	**0,23**	0,28
		STD	0,25	0,31	0,27	0,28	0,27	**0,26**	0,27
	F-Wert	Micro AVG	0,36	0,57	0,35	0,40	0,48	**0,43**	0,52
		Macro AVG	0,29	0,32	0,27	0,31	0,32	**0,31**	0,36
		STD	0,28	0,32	0,28	0,29	0,30	**0,28**	0,29

[a] Berechnete Werte sind ProP, ProR und ProFW.
[b] Parameterwerte für Triple-S: $\omega_1 = 0,28; \omega_2 = 0,27; \omega_3 = 0,05; \omega_4 = 0,4; \theta = 0,77$.
[c] Parameterwerte für Triple-S2: $\omega_1 = 0,55; \omega_2 = 0,45; \omega_3 = 0,6; \omega_4 = 0,4; \theta_1 = 0,75; \theta_2 = 0,82$.
[d] Für den NHM-Ansatz wurden übermäßig viele Matches berechnet, die wenig sinnvoll erschienen, sodass keine Kennzahlen berechnet wurden.

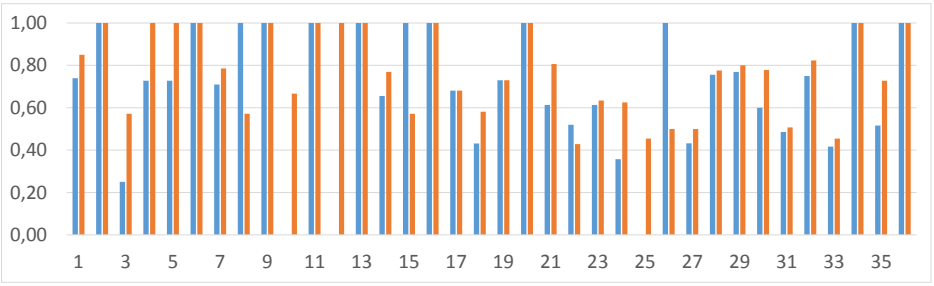
Bei einem Vergleich der Werte des Gesamtstandards mit dem Mehrheitsstandard wird ersichtlich, dass die Precision-Werte jeweils geringer und die Recall-Werte jeweils größer sind. Dies lässt sich darauf zurückführen, dass die beiden Triple-S Verfahren vergleichsweise viele der häufig genannten Matches erkennen, dafür jedoch wenige der seltenen Matches. Dies führt zu einem höheren Recall für den Mehrheitsstandard, da in diesem die Matches mit weniger als drei Nennungen nicht mehr vorhanden sind. Dafür verringert sich der Precision-Wert, da die im Gesamtstandard korrekten, seltenen Matches im Mehrheitsstandard False Positives darstellen. Insgesamt sind jedoch die Ergebnisse für den Mehrheitsstandard besser als für den Gesamtstandard, was sich an den größeren F-Werten erkennen lässt.

Das NLM-Verfahren übertrifft die Triple-S Ansätze in Bezug auf den Precision-Wert, erzielt jedoch im Vergleich niedrigere Recall- und F-Werte. So wird beispielsweise für den Micro AVG Gesamtstandard ein Wert von 0,91 erreicht, während der entsprechende Recall-Wert lediglich bei 0,28 liegt. Aufgrund der geringen Recall-Werte schneidet das NLM-Verfahren auch schlechter bei den berechneten F-Werten ab. Ansonsten treffen die zuvor für die Triple-S Verfahren gemachten Aussagen auch auf den NLM-Ansatz zu.

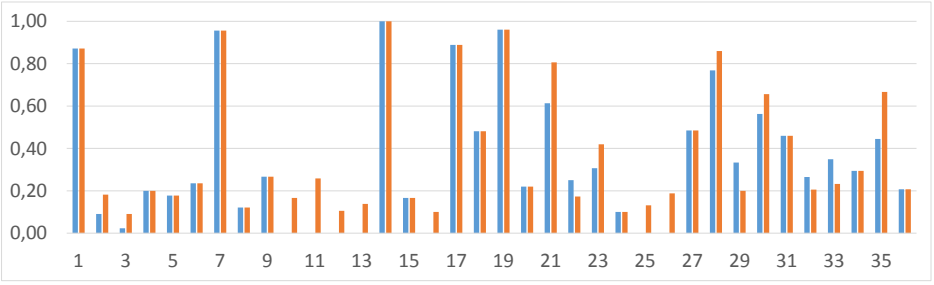
Generell lässt sich zudem feststellen, dass die Werte der Kennzahlen für den University Admission Datensatz relativ gering sind. So bedeutet etwa der Recall-Wert von 0,43 für den Gesamtstandard des Triple-S2 Ansatzes, dass 57 % der korrekten Matches nicht gefunden wurden. Bei dem Triple-S Verfahren sind es sogar nur 39 %. Die Precision-Werte sind hingegen insbesondere für das Triple-S2 Verfahren deutlich besser als die Recall-Werte und liegen auf einem ordentlichen Niveau (30 % der Ergebnisse False Positives). Dieses Ergebnis wird von dem NLM-Verfahren mit einem Höchstwert von 0,91 sogar noch deutlich übertroffen, sodass alle Ansätze tendenziell Schwierigkeiten haben korrekte Matches zu erkennen (niedriger Recall), die ermittelten Matches jedoch meist zutreffend sind (gute bis sehr gute Precision).

Darüber hinaus lässt sich exemplarisch anhand von Abbildung 4.4 erkennen, dass bei beiden Triple-S Verfahren große Unterschiede zwischen den einzelnen Modellpaaren auftreten. Sowohl für die probabilistische Precision als auch für den probabilistischen Recall finden sich Modellpaare mit dem optimalen Wert von 1. Allerdings treten auch sehr geringe Werte auf, beim Triple-S Ansatz insbesondere teilweise der schlechteste Wert 0. Dies resultiert letztlich in den höheren Werten der Standardabweichung in

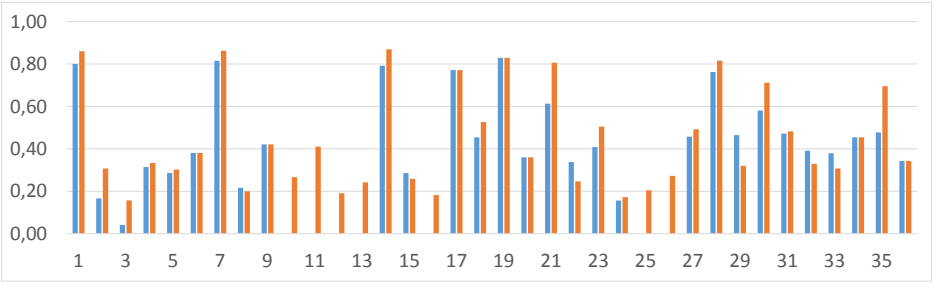
Tabelle 4.4. Anhand dieser Abbildung wird des Weiteren deutlich, dass das Triple-S2 Verfahren gegenüber dem Triple-S Ansatz bei acht Modellpaaren überhaupt erst korrekte Ergebnisse berechnet und sich somit ein Recall-Wert größer 0 ergibt.



(a) Probabilistische Precision



(b) Probabilistischer Recall



(c) Probabilistischer F-Wert

Abb. 4.4: Einzelergebnisse der Triple-S Verfahren für den Mehrheitsstandard des University Admission Datensatzes. Jede Spalte entspricht einem der 36 unterschiedlichen Modellpaare. Blaue Balken stellen die Triple-S Werte dar, orangene Balken geben die Triple-S2 Werte an.

Allerdings ist das Triple-S2 Verfahren bei keiner der drei betrachteten Kennzahlen durchgängig besser als der Triple-S Ansatz. So ist etwa der probabilistische F-Wert der Modellpaare 15, 22, 32 und 33 in Abbildung 4.4c bei dem Triple-S Ansatz höher. Dies trifft auch für einige Modellpaare bezüglich der probabilistischen Precision und dem probabilistischem Recall zu. Daher ergibt sich aus diesem Datensatz, dass sich das Triple-S Verfahren nicht durch Triple-S2 ersetzen lässt, obwohl letzteres im Durchschnitt bessere Ergebnisse berechnet.

Diese beispielhaften Ausführungen zum Verlauf des Mehrheitsstandards gelten auch für die anderen Standards, weshalb die Ergebnisse dieser Goldstandards nicht detaillierter erläutert werden. Die einzelnen Ergebnisse für alle Modellpaare aller Standards finden sich in den Tabellen A.1 bis A.7 in Anhang A.1.

Die Ergebnisse des Birth Registration Datensatzes sind in Tabelle 4.5 aufgelistet. Für diesen Datensatz gelten prinzipiell die gleichen Ausführungen wie zuvor für den University Admission Datensatz. Auch hier ist das Triple-S2 Verfahren besser als der Triple-S Ansatz. Allerdings ist der Abstand zwischen den beiden Verfahren nicht so groß wie bei dem UA-Datensatz. Die Verbesserung der F-Werte des Triple-S2 Verfahrens lassen sich vor allem auf eine gesteigerte Precision zurückführen, während die Recall-Werte nahezu gleich sind.

Zudem sind die Werte für den Birth Registration Datensatz generell höher als für die UA-Modelle. Vor allem für den Mehrheitsstandard werden gute Ergebnisse erzielt. Allerdings fallen auch bei diesem Datensatz die geringen Recall-Werte der beiden Triple-S Verfahren auf, die für den Gesamtstandard beinahe gleich sind wie für den University Admission Datensatz. Bei den einzelnen Standards ist auffällig, dass sich die einzelnen Recall-Werte stärker unterscheiden. So beträgt für das Triple-S2 Verfahren der Wert für Goldstandard 2 lediglich 0,24, während Goldstandard 5 einen Wert von 0,80 aufweist. Die Werte der restlichen drei Standards liegen im Intervall [0,44, 0,56]. Die Precision-Werte liegen demgegenüber wesentlich enger zusammen im Intervall [0,67, 0,76].

Das NLM-Verfahren erzielt ebenso wie für den University Admission Datensatz sehr gute Precision-Werte von bis zu 0,96. Allerdings sind die Recall-Werte ebenfalls wieder wesentlich geringer (Maximalwert von 0,56), sodass sich geringere F-Werte als bei den Triple-S Ansätzen ergeben. Das NHM-Verfahren schneidet hingegen insbesondere bei den Recall-Werten gut ab, wobei für den Mehrheitsstandard der höchste Werte für den Micro AVG von 0,80 erreicht wird. Erkennbar wird zudem, dass der

Tab. 4.5: Ergebnisse der Matching-Verfahren für den Birth Registration Datensatz

			GS1	GS2	GS3	GS4	GS5	Gesamt[a]	Mehrheit[a]
Triple-S[b]	Precision	Micro AVG	0,70	0,71	0,62	0,62	0,59	**0,82**	0,76
		Macro AVG	0,80	0,79	0,70	0,59	0,68	**0,85**	0,78
		STD	0,24	0,24	0,30	0,35	0,28	**0,20**	0,24
	Recall	Micro AVG	0,53	0,24	0,56	0,44	0,78	**0,43**	0,68
		Macro AVG	0,40	0,22	0,52	0,31	0,79	**0,32**	0,63
		STD	0,29	0,24	0,36	0,31	0,30	**0,27**	0,34
	F-Wert	Micro AVG	0,60	0,36	0,59	0,51	0,67	**0,56**	0,72
		Macro AVG	0,45	0,28	0,51	0,32	0,63	**0,40**	0,63
		STD	0,27	0,24	0,32	0,27	0,29	**0,27**	0,31
Triple-S2[c]	Precision	Micro AVG	0,74	0,76	0,67	0,67	0,67	**0,87**	0,80
		Macro AVG	0,84	0,85	0,75	0,65	0,74	**0,90**	0,82
		STD	0,24	0,23	0,31	0,36	0,28	**0,19**	0,24
	Recall	Micro AVG	0,52	0,24	0,56	0,44	0,80	**0,43**	0,69
		Macro AVG	0,39	0,22	0,52	0,31	0,80	**0,32**	0,63
		STD	0,28	0,23	0,36	0,30	0,30	**0,26**	0,34
	F-Wert	Micro AVG	0,61	0,36	0,61	0,53	0,72	**0,57**	0,74
		Macro AVG	0,45	0,29	0,53	0,33	0,67	**0,41**	0,65
		STD	0,28	0,25	0,33	0,30	0,30	**0,27**	0,31
NHM	Precision	Micro AVG	0,57	0,65	0,51	0,51	0,49	**0,76**	0,65
		Macro AVG	0,50	0,66	0,42	0,41	0,45	**0,72**	0,53
		STD	0,24	0,20	0,24	0,29	0,29	**0,23**	0,26
	Recall	Micro AVG	0,62	0,32	0,67	0,52	0,93	**0,52**	0,80
		Macro AVG	0,51	0,33	0,64	0,44	0,95	**0,43**	0,74
		STD	0,25	0,21	0,33	0,32	0,12	**0,24**	0,32
	F-Wert	Micro AVG	0,60	0,43	0,58	0,52	0,64	**0,62**	0,71
		Macro AVG	0,47	0,39	0,46	0,38	0,55	**0,50**	0,57
		STD	0,22	0,19	0,24	0,27	0,27	**0,21**	0,24
NLM	Precision	Micro AVG	0,83	0,86	0,77	0,76	0,80	**0,92**	0,96
		Macro AVG	0,95	0,96	0,87	0,76	0,92	**0,96**	0,76
		STD	0,23	0,21	0,35	0,36	0,27	**0,10**	0,36
	Recall	Micro AVG	0,40	0,19	0,45	0,34	0,67	**0,34**	0,56
		Macro AVG	0,28	0,16	0,42	0,21	0,62	**0,23**	0,52
		STD	0,25	0,21	0,37	0,25	0,38	**0,24**	0,38
	F-Wert	Micro AVG	0,54	0,31	0,57	0,47	0,73	**0,50**	0,67
		Macro AVG	0,35	0,22	0,46	0,26	0,65	**0,32**	0,57
		STD	0,28	0,24	0,38	0,29	0,39	**0,27**	0,38

[a] Berechnete Werte sind ProP, ProR und ProFW.
[b] Parameterwerte für Triple-S: $\omega_1 = 0,28; \omega_2 = 0,27; \omega_3 = 0,05; \omega_4 = 0,4; \theta = 0,77$.
[c] Parameterwerte für Triple-S2: $\omega_1 = 0,55; \omega_2 = 0,45; \omega_3 = 0,6; \omega_4 = 0,4; \theta_1 = 0,75; \theta_2 = 0,82$.

NHM-Ansatz vergleichsweise viele der selten in den Goldstandards vorkommenden Matches erkennt, da die Recall-Werte für den Gesamtstandard deutlich über den Werten der anderen Verfahren liegen. Die Precision-Werte sind jedoch geringer als bei den anderen verglichenen Ansätzen. So ergeben sich für NHM höhere F-Werte in Bezug auf den Gesamtstandard im Vergleich zu den Triple-S Verfahren, diese weisen dagegen bessere F-Werte für den Mehrheitsstandard auf.

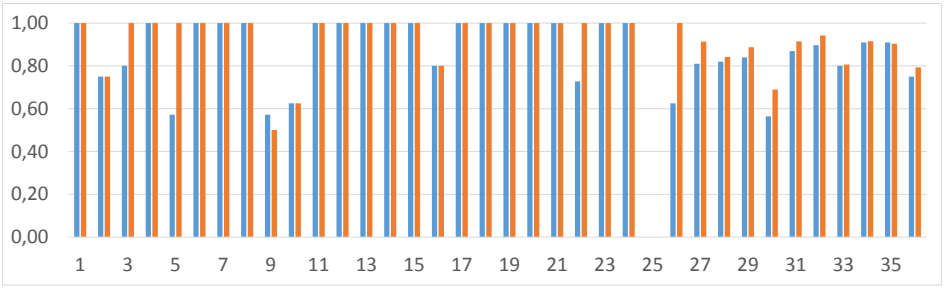
Darüber hinaus gelten für den Birth Registration Datensatz die exemplarischen Ausführungen zum Verlauf des Mehrheitsstandards ebenso für die anderen Standards, weshalb diese nicht detaillierter erläutert werden. Die Resultate der Triple-S Verfahren für alle Modellpaare aller Standards finden sich in den Tabellen A.8 bis A.14 in Anhang A.1.

Bei genauerer Untersuchung der einzelnen Matching-Ergebnisse für den Mehrheitsstandard des Birth Registration Datensatzes (Abbildung 4.5) ergibt sich ein anderes Bild im Vergleich zu dem University Admission Mehrheitsstandard. Im Gegensatz zu den UA-Modellen finden sich bei den BR-Modellen kaum Unterschiede zwischen den beiden Triple-S Verfahren. Der Triple-S2 Ansatz ist teilweise besser in Bezug auf die Precision-Werte, was sich auch im Gesamtergebnis widerspiegelt, allerdings sind die Recall-Werte beinahe gleich. Vor allem erkennt das Triple-S2 Verfahren für vier Modellpaare keine korrekten Matches, sodass der Recall-Wert bei 0 liegt. Hier war der Triple-S2 Ansatz bei dem University Admission Datensatz besser, da dabei für alle Modellpaare korrekte Matches berechnet wurden.

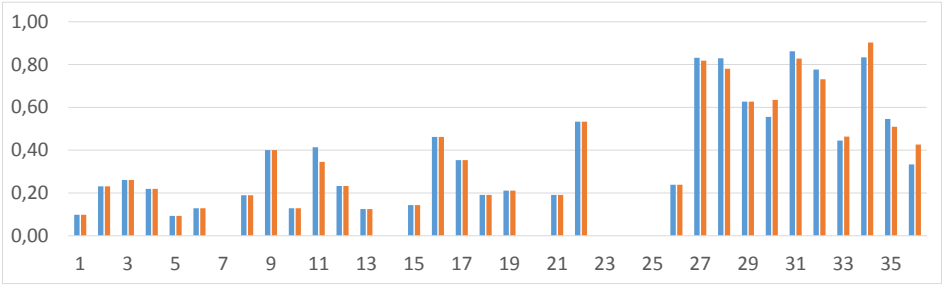
4.4.3 Diskussion

Im Folgenden werden fünf Aspekte in Bezug auf die entwickelten Triple-S Verfahren und die Evaluation detaillierter erläutert. Zum einen werden die Qualität der Ergebnisse und offene Herausforderungen beschrieben. Zum anderen wird auf die Anzahl der Modelle und Goldstandards als einschränkende Faktoren der Evaluation eingegangen sowie eine mögliche Übertragbarkeit der Triple-S Verfahren auf Modelle in anderen Modellierungssprachen diskutiert. Schließlich werden die Ergebnisse dieser Evaluation in Beziehung gesetzt zu den Resultaten des Process Model Matching Contest 2015 (Antunes u. a. 2015).

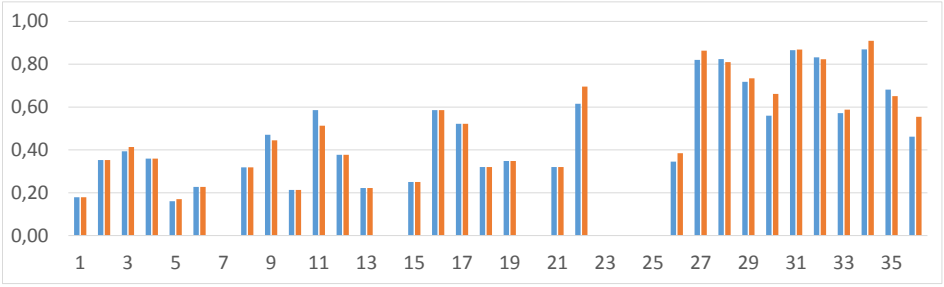
Qualität der Ergebnisse: Hinsichtlich der Qualität der Ergebnisse für die Gesamtstandards zeigt sich, dass beide Triple-S Verfahren relativ gute Precision-Werte für bei-



(a) Probabilistische Precision



(b) Probabilistischer Recall



(c) Probabilistischer F-Wert

Abb. 4.5: Einzelergebnisse der Triple-S Verfahren für den Mehrheitsstandard des Birth Registration Datensatzes. Jede Spalte entspricht einem der 36 unterschiedlichen Modellpaare. Blaue Balken stellen die Triple-S Werte dar, orangene Balken geben die Triple-S2 Werte an.

de Datensätze erzielen. Insbesondere der Triple-S2 Ansatz erreicht einen guten Wert von 0,87 für die Birth Registration Modelle. Die Recall-Werte sind dagegen vergleichs-

weise gering. Der höchste Wert beträgt lediglich 0,43, sodass mehr als die Hälfte der korrekten Matches nicht gefunden wurden. Dies resultiert in den jeweils durchschnittlichen F-Werten von ca. 0,55.

Bezüglich der Mehrheitsstandards lässt sich für die beiden evaluierten Datensätze feststellen, dass die Precision-Werte sinken und die Recall-Werte steigen. Dies liegt daran, dass in den Mehrheitsstandards weniger Matches vorhanden sind (alle Matches, die nur ein- oder zweimal von den Experten genannt wurden, wurden entfernt), die jedoch überwiegend von den Triple-S Verfahren erkannt wurden. Dadurch steigen die Recall-Werte, die Precision-Werte sinken hingegen, weil einige der für den Gesamtstandard korrekten Matches nun False Positives sind. Für den Birth Registration Datensatz steigen die Recall-Werte deutlich, während sich die Precision-Werte nur geringfügig verringern, sodass sich ein guter F-Wert von 0,74 ergibt. Für die University Admission Modelle ist dies jedoch nicht in diesem Ausmaß ersichtlich, sodass zwar deutlich wird, dass die von den Experten eher unstrittig bestimmten Matches besser erkannt werden, sehr gute Ergebnisse werden jedoch nicht erreicht.

Herausforderungen bei der Match-Berechnung: Bei einer genaueren Betrachtung der Evaluationsergebnisse konnten die folgenden fünf Problemklassen für die geringen Recall-Werte ermittelt werden. Diese beziehen sich auf die Beschriftungsweise sowie die Art der Modellierung. Beispiele zu jeder Problemklasse von nicht erkannten Matches aus dem Birth Registration Datensatz sind in Tabelle 4.6 aufgelistet.

Tab. 4.6: Problemklassen bei der Match-Berechnung

Problemklasse	Modellpaar	Beispiel
Ungebräuchliche Synonyme	p31 – p32	Check data (t1) – Check documents (t41)
Unbekannte Abkürzungen	p246 – p248	Receive notification birth (t1) – Receive notific. of birth (t1)
Restriktive Schwellenwerte	p246 – p247	Check GBA data (t9) – Check GBA (t7)
Unterschiedliche Modellierungsweise	p31 – p33	Mother German? (t11), Father German? (t13) – Check parent's nationality (t11)
Detaillierungsgrad von Modellen	p247 – p250	Check GBA (t7) – Check GBA (t12)

Die Klasse *ungebräuchliche Synonyme* bezieht sich auf Wörter, die nicht als Synonyme erkannt werden. In dem Beispiel aus Tabelle 4.6 ergibt sowohl die Wortähnlichkeit nach (Wu und Palmer 1994) als auch nach (Mikolov u. a. 2013b) einen zu geringen Wert für „data“ und „documents“, sodass die Ähnlichkeit der Beschriftungen ins-

gesamt zu gering ist, um ein Match der beiden Transitionen zu ergeben. Dieses Problem tritt zudem auf, wenn Wörter in Beschriftungen verwendet werden, für die keine Ähnlichkeitswerte berechnet werden können, da sie nicht in WORDNET vorkommen oder nicht durch das WORD2VEC-Verfahren bestimmt wurden. Als Lösungsoptionen könnten die Wörter WORDNET hinzugefügt werden bzw. die Wortähnlichkeiten über Dokumente, die diese Begriffe enthalten, von WORD2VEC gelernt werden.

Bei dem zweiten Problem handelt es sich um *unbekannte Abkürzungen*, für die ebenfalls keine Wortähnlichkeiten berechnet werden können. Obwohl die Beschriftungen die gleiche Aktivität beschreiben, werden die Transitionen aufgrund der Abkürzung von „notification“ nicht erkannt.[16] Eine Lösung hierfür wäre die Auflösung von Abkürzungen durch ihre ausgeschriebene Form, was auch automatisch durchgeführt werden könnte, wenn etwa eine entsprechende Liste vorläge.

Die dritte Problemklasse *restriktive Schwellenwerte* ergibt sich durch die relativ hohen Schwellenwerte, ab denen ein Match vorliegt. Die hohen Schwellenwerte führen zu den vergleichsweise guten Precision-Werten, haben jedoch zur Folge, dass etwa die Transitionen „Check GBA data“ und „Check GBA“ nicht als Match erkannt werden, obwohl sie sich auf die gleiche Aktivität beziehen. Durch das zusätzliche Wort „data“ ergibt sich ein zu geringer Ähnlichkeitswert der Beschriftungen sowohl auf syntaktischer als auch auf semantischer Ebene. Hier existiert keine offensichtliche Lösung, da eine Verringerung von Schwellenwerten zu einer geringeren Precision führen würde. Eventuell würde eine detailliertere Betrachtung der Beschriftungen dieses Problem lösen, durch die erkannt wird, dass zwei Wörter der Beschriftungen identisch sind und die eine Beschriftung damit vollständig in der anderen enthalten ist.

Die vierte Klasse erzeugt durch eine *unterschiedliche Beschriftungs- und Modellierungsweise* die größten Schwierigkeiten. In dem Beispiel aus Tabelle 4.6 werden statt einer Transition zur Modellierung der Nationalität von Eltern zwei Transitionen verwendet, die darüber hinaus eine gänzlich unterschiedliche Wortwahl in den Beschriftungen aufweisen. Das Wissen, dass Mutter und Vater Eltern sind und dass Deutsch eine mögliche Nationalität ist, wird nicht über Wortähnlichkeiten abgebildet, sodass für solche Fälle zusätzliche Ähnlichkeitsberechnungen benötigt werden (z. B. spezifische Ontologien, in denen die Beziehungen zwischen Mutter, Vater und Eltern beschrieben sind). Erschwert werden solche Fälle zudem durch die Aufteilung einer Aktivität in

[16] Das Stoppwort „of“ wird bei dem Vergleich der Beschriftungen nicht berücksichtigt.

zwei Aktivitäten. Die Triple-S Verfahren betrachten jeweils nur Paare von Transitionen, sodass kein gemeinsamer Vergleich der drei Transitionen erfolgt.

Bei der fünften Problemklasse ist schließlich der *Detaillierungsgrad von Modellen* zu unterschiedlich, sodass sich zu stark differierende Werte für die relative Position von Transitionen ergibt. In dem Beispiel in Tabelle 4.6 sind beide Transitionen gleich beschriftet, jedoch ist der Detaillierungsgrad der beiden Modelle sehr unterschiedlich. In diesem Fall sind die Aktivitäten in Modell p250, die vor „Check GBA" ausgeführt werden, ausführlicher modelliert, d. h. durch mehr Transitionen. Eine Lösung könnte die Verwendung einer Heuristik sein, dass Transitionen mit syntaktischer Ähnlichkeit von 1 immer ein Match ergeben, unabhängig von der relativen Position.

Anzahl an Modellen und Goldstandards: Eine Einschränkung der Evaluation existiert hinsichtlich der Anzahl an Modellen und Goldstandards. Bei der Evaluation wurden lediglich zwei kleine Modelldatensätze mit jeweils neun Modellen eines Prozesses verwendet, die separat verglichen wurden, d. h., die Modelle der beiden Prozesse wurden nicht zu einem Datensatz zusammengefügt. Somit können keine Aussagen darüber getroffen werden, wie gut die Ansätze bei großen Modelldatensätzen abschneiden, in denen insbesondere Modelle verschiedener Prozesse vorhanden sind.

Für die Triple-S Verfahren sind keine Veränderungen bezüglich der Ergebnisqualität zu erwarten, wenn Modelle verschiedener Prozesse in einem Datensatz vorhanden sind. Die Triple-S Ansätze vergleichen jeweils Modellpaare separat und damit unabhängig von den anderen Modellen in einem Datensatz. Hierbei wäre jedoch interessant, wie gut das RefMod-Mine/NHM Verfahren von Thaler et al. aus (Antunes u. a. 2015) abschneidet, da dabei alle Aktivitäten aller Modelle gemeinsam durch ein Clusterverfahren verglichen werden. Des Weiteren wurde noch nicht speziell überprüft, inwieweit die Triple-S Verfahren einzelne Aktivitäten, die in unterschiedlichen Prozessen vorkommen (z. B. eine Aktivität „Kunde kontaktieren", die in verschiedenen Prozessen vorkommen kann), erkennen können. Auch hier wird erwartet, dass sich die Triple-S Verfahren nicht anders verhalten als in der beschriebenen Evaluation und damit die gleichen Schwierigkeiten haben, wie im vorigen Punkt erläutert. Diese beiden Aspekte sind jedoch bislang nicht explizit überprüft worden.

Darüber hinaus wurden lediglich die Goldstandards von fünf Modellierungsexperten zur Evaluation herangezogen. Eine größere Anzahl von Standards wäre hilfreich, um etwa klarer die eindeutigen Matches in den Mehrheitsstandards von den uneindeutigen durch ihre Häufigkeit unterscheiden zu können. Interessant wäre zudem, ob die

Standards von Prozessausführenden und Modellierungsexperten differieren würden und sich somit auch eine andere Ergebnisqualität ergeben würde. Die Berücksichtigung dieser Aspekte war jedoch nicht möglich, da zum einen die manuelle Erstellung der Goldstandards sehr zeitaufwändig ist[17] und zum anderen für die verwendeten Modelle keine Prozessteilnehmer zur Verfügung standen.

Übertragung auf andere Modellierungssprachen: Die Triple-S Verfahren können auf Prozessmodelle in anderen imperativen Modellierungssprachen übertragen werden, um beispielsweise die Sprachelemente aufeinander abzubilden, die Aktivitäten entsprechen. Dies wurde im Rahmen des Matching Wettbewerbs 2015 durch das Auffinden von ähnlichen Aktivitäten in EPKs und BPMN-Modellen für den Triple-S Ansatz demonstriert (Antunes u. a. 2015). Prinzipiell müssen dazu entsprechende Matches wie in Definition 4.1 für die jeweilige Modellierungssprache festgelegt werden. Des Weiteren müssen die Modelle in anderen Modellierungssprachen ebenfalls die Restriktionen von Workflow-Netzen einhalten, d. h. genau einen Start- und genau einen Endknoten besitzen. Dies ist notwendig für die Berechnung der relativen Position von Aktivitäten.

Die folgenden exemplarischen Ausführungen für *Ereignisgesteuerte Prozessketten* (Keller u. a. 1992) sollen die Übertragbarkeit verdeutlichen, ohne dass eine formale Beschreibung gegeben wird. EPKs enthalten Ereignisse und Funktionen, die Ereignisse im Prozessablauf bzw. Aktivitäten darstellen. Des Weiteren wird der Ablauf eines Prozesses in EPKs durch Kontrollflusskonnektoren dargestellt. Darauf aufbauend kann z. B. die in diesem Kapitel verwendete Definition von Matches für Transitionen auf Funktionen in EPKs übertragen werden, da beide Modellierungskonstrukte Aktivitäten repräsentieren. Um die Restriktionen von Workflow-Netzen einzuhalten, müsste zudem festgelegt werden, dass die EPKs jeweils mit einem Ereignis beginnen und mit einem Ereignis enden. Dadurch kann die relative Position von Funktionen in EPKs bestimmt werden, wobei die expliziten Kontrollflusskonnektoren nicht gesondert berücksichtigt werden.

Prozessmodelle in deklarativen Sprachen können hingegen nicht direkt durch die Triple-S Verfahren verglichen werden, da sich die relative Positionsberechnung nicht durchführen lässt und die Anzahl an ein- bzw. ausgehenden Kanten nicht bestimmbar

[17] Zur Erstellung eines Goldstandards für einen der Evaluationsdatensätze wurden von jedem Experten mehrere Stunden benötigt.

ist. Deklarative Prozessmodelle beschreiben Aktivitäten, die ausgeführt werden können, sowie Restriktionen, die nicht erwünschtes Verhalten einschränken sollen (Haisjackl u. a. 2013). Abbildung 4.6 zeigt ein Beispiel eines deklarativen Prozessmodells. Die in dem Prozessmodell enthaltenen Aktivitäten sind in dem Kasten oben links dargestellt, die Restriktionen zur Einschränkung nicht erwünschter Abfolgen von Aktivitäten sind in dem linken unteren Kasten abgebildet. Daraus wird ersichtlich, dass solche Modelle nicht notwendigerweise zusammenhängend sind und unterschiedliche Arten von Kanten enthalten können. Somit sind die strukturellen Berechnungen der Triple-S Verfahren nicht durchführbar.

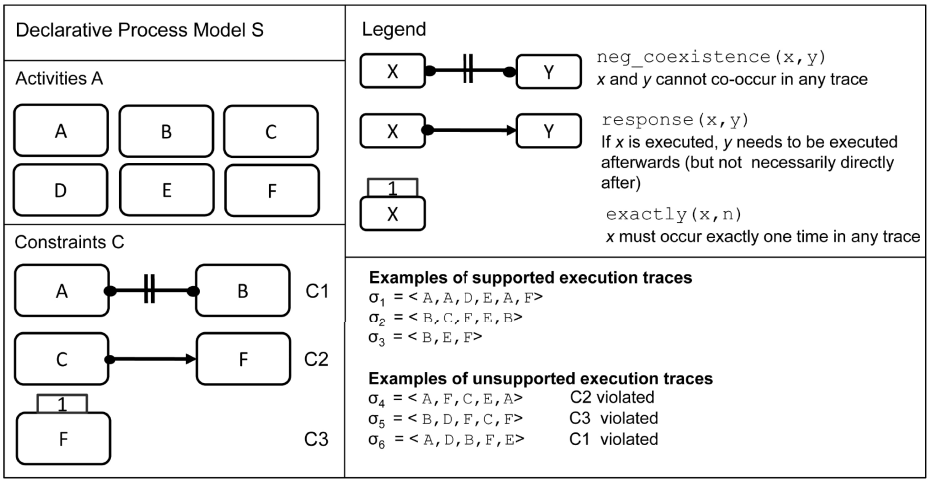


Abb. 4.6: Beispiel eines deklarativen Prozessmodells aus (Haisjackl u. a. 2013)

Process Model Matching Contest 2015: Die Ergebnisse der zuvor beschriebenen Evaluation lassen sich nicht direkt mit den Resultaten des Process Model Matching Contest 2015 (Antunes u. a. 2015) vergleichen, da weitere Goldstandards verwendet wurden. Allerdings zeigt die Evaluation ähnliche Tendenzen bei einer Betrachtung der F-Werte für den Gesamtstandard. So schneidet das REFMOD-MINE/NHM Verfahren für den Birth Registration Datensatz am besten von allen verglichenen Ansätzen ab, was auch in dem Matching Contest 2015 der Fall war. Die Triple-S Verfahren folgen und der REFMOD-MINE/NLM Ansatz schneidet am schlechtesten für diesen Datensatz ab, was ebenfalls der Reihenfolge in dem Matching Contest 2015 entspricht. Für den Mehrheitsstandard ergibt sich eine andere Reihenfolge. In diesem Fall liegen die Triple-S Verfahren knapp vor dem REFMOD-MINE/NHM Ansatz.

Für den University Admission Datensatz zeigt sich ein anderes Resultat. Während in dem Matching Contest 2015 beide REFMOD-MINE Verfahren vor dem Triple-S Ansatz lagen, ist dies nun umgekehrt. Die beiden Triple-S Verfahren erreichen höhere F-Werte als der REFMOD-MINE/NLM Ansatz. Für REFMOD-MINE/NHM wurden keine Ergebnisse berechnet, da über die RefMod-Miner Plattform eine Vielzahl an Matches berechnet wurden, die nicht nachvollziehbar waren, sodass von einer Berechnung der Kennzahlen abgesehen wurde.

4.5 Zusammenfassung

In diesem Kapitel wurden zwei Ansätze beschrieben, mit denen ähnliche Aktivitäten in Prozessmodellen ermittelt werden können. Die beiden Triple-S Verfahren zum Prozessmodell-Matching berechnen dabei die Ähnlichkeit von Aktivitäten (Transitionen) anhand verschiedener Dimensionen von Prozessmodellen. So berücksichtigen beide Verfahren die String-Editier Distanz (als syntaktische Ebene bezeichnet) sowie die semantische Ähnlichkeit der Beschriftungen von Transitionen. Die semantische Ähnlichkeit wird bei Triple-S durch das Verfahren von Wu und Palmer (1994) zur Bestimmung von Wortähnlichkeiten bestimmt, bei Triple-S2 wird hingegen auf Wortähnlichkeiten zurückgegriffen, die mit Hilfe von WORD2VEC (Mikolov u. a. 2013b) berechnet wurden.

Neben der natürlichsprachlichen Dimension wird die Graphstruktur von Prozessmodellen zur Ähnlichkeitsberechnung genutzt. Für das Triple-S Verfahren werden dazu die Anzahl an eingehenden und ausgehenden Kanten von Transitionen sowie die relative Position innerhalb eines Modells miteinander verglichen. Für den Triple-S2 Ansatz wird lediglich die relative Position verglichen. Zudem werden bei Triple-S2 die syntaktische und semantische Ähnlichkeit nicht kombiniert berücksichtigt wie noch bei dem Triple-S Ansatz, sondern jeweils getrennt.

Bei der Evaluation beider Verfahren hat sich gezeigt, dass Triple-S2 geringfügig bessere F-Werte erreicht als Triple-S. Dabei erreichen beide Ansätze gute Precision-Werte, allerdings nur mittelmäßige Recall-Werte. Im Vergleich mit zwei in (Antunes u. a. 2015) publizierten Verfahren schnitten die Triple-S Ansätze besser ab als REFMOD-MINE/NLM, jedoch schlechter als REFMOD-MINE/NHM. Allerdings erzielt keines

der Matching-Ansätze bislang zufriedenstellende Resultate, da der höchste ermittelte F-Wert lediglich 0,74 beträgt. Eine grundsätzliche Herausforderung bei der Evaluation ist darüber hinaus die Erstellung eines repräsentativen Goldstandards. Die von fünf Modellierungsexperten erstellten Standards enthielten viele unterschiedliche Matches, die für die automatisierten Matching-Verfahren teilweise schwierig zu ermitteln waren. Für die von einer Mehrheit der Experten genannten Matches schnitten die Triple-S Ansätze dagegen deutlich besser ab.

Kapitel 5

Ähnlichkeitsbasierte Suche mit Prozessmodellen

In diesem Kapitel wird eine ähnlichkeitsbasierte Suchfunktion für Geschäftsprozessmodellbibliotheken beschrieben. Dabei wird ein Prozessmodell als ein sogenanntes Anfragemodell verwendet, um zu diesem Modell ähnliche Modelle zu finden, die einen gewissen Mindestgrad an Ähnlichkeit aufweisen. Diese Suchfunktion beruht nicht wie viele verwandte Arbeiten auf einem Prozessmodell-Matching, da das Auffinden korrekter Matches sich wie in Kapitel 4 beschrieben als schwierig herausgestellt hat. Stattdessen stellt das *LS3: Latent Semantic Analysis-based Similarity Search* Verfahren die textuellen Beschriftungen von Modellelementen in ihrer Gesamtheit in den Mittelpunkt. Dies bedeutet, dass im Gegensatz zur Ähnlichkeitsberechnung mit Hilfe von Matches nicht einzelne sich entsprechende Aktivitäten als Grundlage zur Ähnlichkeitsberechnung verwendet werden, sondern dass die Beschriftungen von Prozessmodellelementen als textuelle Beschreibung eines Prozessablaufs aufgefasst und diese miteinander verglichen werden. Dieses Verfahren basiert auf den in (Schoknecht u. a. 2017a) und (Schoknecht und Oberweis 2017) beschriebenen Ausführungen.

Im Folgenden wird zunächst in Kapitel 5.1 eine formale Problembeschreibung eingeführt, um anschließend auf verwandte Arbeiten einzugehen. In Kapitel 5.3 werden die entwickelten Suchfunktionen beschrieben, die anschließend in Kapitel 5.4 evaluiert werden. Schließlich werden in Abschnitt 5.5 die Erkenntnisse dieses Kapitels zusammengefasst.

5.1 Problembeschreibung

Die in diesem Kapitel beschriebene Suchfunktion ermöglicht es, zu einem Prozessmodell q (einem sogenannten *Anfragemodell*) ähnliche Prozessmodelle in einer Prozessmodellbibliothek **M** zu finden. Diese Modelle weisen einen Ähnlichkeitsgrad auf, der gleich groß ist wie ein Schwellenwert θ oder über diesem liegt. Das zur Berechnung eines Ähnlichkeitswerts entwickelte Ähnlichkeitsmaß *Latent Semantic Analysis-based Similarity Measure* (LSSM) (Schoknecht und Oberweis 2017) verwendet die in Kapitel 3.3 erläuterte Latent Semantic Analysis. Diese analysiert ursprünglich die Wortwahl von Textpassagen einer gegebenen Länge, sogenannten *Dokumenten* bzw. *Dokumentvektoren*. Demzufolge muss zunächst eine entsprechende Dokumentenform für die verwendeten Prozessmodelle und ihre Beschriftungen definiert werden. Dazu wird festgelegt, dass ein Prozessmodell als Basis eines Dokuments verwendet wird. Ein Dokumentenvektor eines Prozessmodells ist daher folgendermaßen definiert:

Definition 5.1: Dokumentenvektor eines Prozessmodells

Sei W_{all} eine Menge von Termen, die alle eindeutigen Terme einer Prozessmodellbibliothek enthält. **M** sei eine Menge von Prozessmodellen (die Prozessmodellbibliothek) und $w(m)$ sei eine Funktion, die die Menge aller Terme (*Bag-of-Words*) W_m eines Prozessmodells $m \in \mathbf{M}$ zurückgibt mit $W_m \subseteq W_{all}$.

Der Vektor $d_m = (w_{1m}, w_{2m}, ..., w_{tm})$ repräsentiert dann den Dokumentenvektor des Prozessmodells m, wobei jeder Index i einen Term der Menge aller Terme darstellt, die in der Prozessmodellbibliothek enthalten sind: $W_{all} = \bigcup w(m)$ für alle $m \in \mathbf{M}$. Der Eintrag w_{im} reflektiert das Gewicht der Termhäufigkeit, die beschreibt, wie oft ein bestimmter Term in einem untersuchten Prozessmodell vorkommt (Schoknecht und Oberweis 2017).

Wichtig dabei ist, dass ein Anfragemodell q prinzipiell wie ein Modell einer Prozessmodellbibliothek behandelt wird, d. h., dass q ebenfalls als ein Dokumentenvektor repräsentiert wird. Das LSSM-Ähnlichkeitsmaß berechnet schließlich einen Ähnlichkeitswert zwischen den Vektorrepräsentationen eines Anfragemodells q und allen

Modellen $m \in \mathbf{M}$ einer Prozessmodellbibliothek im Intervall $[0, 1]$.[1] Als Ergebnis werden die Modelle aus $\mathbf{M}$ zurückgegeben, die einen gleich hohen oder größeren Ähnlichkeitswert zu einem Schwellenwert θ aufweisen. Das LS3-Verfahren berechnet somit folgende Ergebnismenge QR:

$$QR = \{m \mid m \in \mathbf{M} \wedge LSSM(q, m) \geq \theta\} \tag{5.1}$$

Im Gegensatz zu dem typischerweise in verwandten Arbeiten vorherrschenden zweistufigen Vorgehen, bei dem zunächst Matches zwischen Prozessmodellen identifiziert werden und darauf aufsetzend ein Ähnlichkeitsgrad von Modellen ermittelt wird (Schoknecht u. a. 2017b), wird bei dem LS3-Verfahren der erste Schritt nicht benötigt. Für die in diesem Kapitel beschriebene ähnlichkeitsbasierte Suchfunktion ist die Berechnung von Matches nicht zwingend erforderlich, da die textuellen Beschriftungen den Inhalt eines Prozesses im Sinne von enthaltenen Aktivitäten, Ereignissen, beteiligten Rollen etc. beschreiben. Folglich wird im Kontext einer Suchfunktion beispielsweise das Ablaufverhalten eines Prozessmodells nicht notwendigerweise für die Ähnlichkeitsberechnung benötigt. Für andere Anwendungsfälle einer Ähnlichkeitsberechnung (siehe Kapitel 2.2.3) muss diese Annahme jedoch nicht gelten. Soll z. B. die Übereinstimmung eines Prozessmodells mit einem zugehörigen Referenzmodell überprüft werden, so ist die Berechnung von Matches erforderlich, um detailliert Unterschiede ermitteln zu können (etwa welche Aktivitäten in dem Prozessmodell gegenüber dem Referenzmodell zusätzlich ausgeführt werden bzw. welche fehlen).

Im Rahmen der angestrebten Förderung der Wiederverwendung von Modellen oder Modellteilen, die durch die beschriebenen Suchverfahren unterstützt werden soll, ist neben der Qualität der Suchergebnisse auch die Berechnungszeit von Relevanz. Für einen Modellierer, der beispielsweise anhand eines bereits bestehenden Modells nach weiteren existierenden ähnlichen Modellen sucht, steht im Vordergrund, dass Suchergebnisse schnell berechnet werden können und dass die gefundenen Modelle eine hohe Ähnlichkeit zu dem Anfragemodell aufweisen, sodass die Modellierungstätigkeit möglichst gut unterstützt wird.

Bei der Bewertung des Ähnlichkeitsgrads von Prozessmodellen stellt sich insbesondere die Wortwahl in Beschriftungen von Prozessmodellelementen als Herausforderung

[1] Für weitere Details zur Quantifizierung der Ähnlichkeit von Prozessmodellen siehe Kapitel 2.2.1.

dar. Untersuchungen verschiedener Modellsammlungen zeigen, dass in Beschriftungen häufig Synonyme und Homonyme verwendet werden (Pittke u. a. 2013; Pittke 2016), sodass eine Suchfunktion bzw. die Ähnlichkeitsbestimmung dies berücksichtigen muss. Dabei treten prinzipiell dieselben Schwierigkeiten auf wie im Kontext des Prozessmodell-Matchings (siehe Kapitel 4). Synonyme erschweren die Ähnlichkeitsberechnung, da etwa für die gleiche Aktivität unterschiedliche Verben zur Beschreibung verwendet werden (z. B. „Antrag versenden“ und „Antrag verschicken“). Homonyme sind dagegen Wörter, die unterschiedliche Bedeutungen haben. Ein Beispiel dafür aus dem Englischen ist das Wort „application“. Im Kontext von Hochschulprozessen kann „application“ für die Bewerbung eines Studierenden stehen, im Kontext von Prozessen in Informationssystemen kann es hingegen zur Bezeichnung eines Computerprogramms verwendet werden. Diese Aspekte sind insbesondere wesentlich für die in Kapitel 5.3 beschriebenen Suchalgorithmen, da sich diese lediglich auf die textuellen Beschriftungen beziehen und die Graphstruktur- und Verhaltensdimensionen für die Ähnlichkeitsberechnung nicht berücksichtigen.

Somit treten bei der Ähnlichkeitsberechnung von Prozessmodellen bezüglich der textuellen Beschriftungen die gleichen Herausforderungen auf, die bereits beim Matching von Aktivitäten ersichtlich wurden. Da jedoch in dem LS3-Verfahren alle Beschriftungen in ihrer Gesamtheit miteinander verglichen werden, wird erwartet, dass die Verwendung einzelner Synonyme oder Homonyme keine großen Auswirkungen auf die Ähnlichkeitsberechnung hat. Das heißt, es wird letztlich davon ausgegangen, dass verschiedene Modelle eines Prozesses insgesamt eine ausreichende Anzahl übereinstimmender Wörter aufweisen, sodass deren Übereinstimmung bzw. Ähnlichkeit trotz teilweise unterschiedlicher Wortwahl ersichtlich wird. Diese Schwierigkeiten sind nicht auf den Bereich der Prozessmodellierung begrenzt, sondern treten auch im klassischen Information Retrieval auf (Deerwester u. a. 1990). Da die Latent Semantic Analysis die zuvor erläuterten Schwierigkeiten in diesem Gebiet angeht, wird sie auch im Folgenden für die Suche ähnlicher Prozessmodelle eingesetzt. Zur Diskussion des Einsatzes von LSA gegenüber anderen Topic Modeling Ansätzen wie beispielsweise der Latent Dirichlet Allocation (Blei u. a. 2003) oder von Word Embedding Techniken wie WORD2VEC (Mikolov u. a. 2013b) siehe Kapitel 5.4.3.

5.2 Verwandte Arbeiten

Hinsichtlich verwandter Arbeiten werden im Folgenden Arbeiten zur Messung der Ähnlichkeit von Prozessmodellen berücksichtigt, die durch das in Kapitel 4.2 beschriebene Vorgehen ermittelt wurden. Auch wenn diese Ansätze zur Berechnung eines Ähnlichkeitsgrads nicht immer explizit für die Anwendung in einer Suchfunktion entwickelt wurden, könnten sie dennoch grundsätzlich dazu eingesetzt werden. Verwandte Arbeiten zum Thema Anfragesprachen für Prozessmodelle werden in Kapitel 6.3 erläutert. Im Generellen unterscheiden sich diese Arbeiten von ähnlichkeitsbasierten Suchverfahren durch die Beschreibung und den Einsatz einer speziellen Anfragesprache. Während bei ähnlichkeitsbasierten Suchverfahren existierende Modelle verwendet werden können, ist dies bei den Anfragesprachen nicht möglich. Zudem sind die im Folgenden aufgeführten Arbeiten untergliedert nach der für die Ähnlichkeitsberechnung fokussierten Dimension (siehe Kapitel 2.2.2).

Graphstruktur: Die Ansätze von Grigori u. a. (2006) versuchen die Suche nach Services zu verbessern (Grigori u. a. 2006; Corrales u. a. 2006; Grigori u. a. 2008; Grigori u. a. 2010). Dabei wird die Graphstruktur von Prozessmodellen zur Ähnlichkeitsberechnung verglichen. Es wird eine Graph-Editier Distanz verwendet, die misst, wie viele Änderungen an einem Graphen notwendig sind, um ihn in den anderen zu transformieren. Die dafür benötigten Matches von Graphknoten werden anhand ihrer Beschriftungen durch einen syntaktischen und semantischen Vergleich ermittelt. Der semantische Vergleich erfolgt über das Auffinden von Synonymen mit Hilfe von WordNet. In (Gater u. a. 2010) wird dieser Ansatz um ein komplexeres Prozessmodell-Matching Verfahren ergänzt, sodass nicht nur $1:1$ sondern auch $m:n$ Matches gefunden werden können.

Zur Verbesserung der Laufzeit des Graph-Matchings wird in (Gater u. a. 2011) eine Technik eingesetzt, die die Graphstruktur von Prozessmodellen reduziert, um die Anzahl an Knoten für den Graph-Matching Algorithmus zu verringern. In (Gater u. a. 2012) wird eine Alternative zur Beschleunigung der Ähnlichkeitsberechnung beschrieben, die auf speziellen Indexstrukturen basiert. Die gleiche Idee wird auch in (Minor u. a. 2007) genutzt. Dabei werden gleiche Knoten über identische Bezeichnungen identifiziert und der Ähnlichkeitswert zweier Modelle ergibt sich als Graph-Editier Distanz auf Basis der eingefügten bzw. gelöschten Knoten und Kanten.

Dijkman u. a. (2009b) verwenden ebenfalls eine Graph-Editier Distanz zur Bestimmung eines Ähnlichkeitswerts. Für das Auffinden übereinstimmender Knoten wird allerdings nur eine syntaktische Editier-Distanz eingesetzt. Darüber hinaus werden vier Algorithmen zur Berechnung der besten Graph-Editier Distanz mit unterschiedlicher Komplexität beschrieben. In (Kunze und Weske 2010) wird dazu eine Index-Struktur vorgeschlagen, um die Ähnlichkeitsberechnung zu beschleunigen. Mit Hilfe dieses Index müssen Modellvergleiche teilweise nicht ausgeführt werden, da bereits vor der Berechnung ersichtlich ist, dass ein Modellpaar eine zu geringe Ähnlichkeit aufweist. Eine weitere Ergänzung zu (Dijkman u. a. 2009b) bezüglich des zugrunde liegenden Prozessmodell-Matchings ist in (Ling u. a. 2014) beschrieben. Das Matching von Prozessmodellelementen wird um weitere Aspekte wie Datenobjekte und Rollen sowie die Erkennung von $m : n$ Matches erweitert.

Auch in (Yan u. a. 2010; Yan u. a. 2012c) wird der in (Dijkman u. a. 2009b) beschriebene Ansatz verwendet, wobei dieser um eine Vorklassifizierung erweitert wird und der Fokus insbesondere auf eine Beschleunigung der Berechnung abzielt. Die Vorklassifizierung dient dazu Modellpaare in die Gruppen „sicher ähnlich“, „potentiell ähnlich“ und „sicher unähnlich“ einzuordnen, um die aufwändige Berechnung der Ähnlichkeit nur für die potentiell ähnlichen Modellpaare durchführen zu müssen. Für die Einordnung werden für die Graphstruktur sogenannte Features definiert, die beispielsweise angeben, welche Knoten Start- bzw. Endknoten sind oder welche Knoten in einer Sequenz von Knoten enthalten sind. Die Ähnlichkeit von zwei Prozessmodellen berechnet sich als das Verhältnis aus übereinstimmenden Features zu allen Features. Das den Feature-Definitionen zugrunde liegende Matching wird über eine Editier-Distanz der Beschriftungen und der Anzahl an Knoten im Vor- und Nachbereich bestimmt. Zusätzlich wird eine Indexstruktur für die Features eingeführt, durch die die Ergebnisberechnung beschleunigt wird, indem nicht alle Features berechnet werden müssen. Für Modellpaare, deren Ähnlichkeit zu hoch ist, um sie als unähnlich zu klassifizieren, aber auch zu niedrig, um sie sicher als ähnlich zu bewerten, wird das Verfahren aus (Dijkman u. a. 2009b) zur Ermittlung eines exakten Werts durchgeführt. Dieses Verfahren wird in (Liu u. a. 2014) noch um ein flexibleres Matching von Features ergänzt, das beispielsweise auch einen Vergleich von Features mit unterschiedlicher Anzahl an Knoten ermöglicht.

Li u. a. (2008) beziehen sich ebenso auf eine Graph-Editier Distanz, allerdings berücksichtigen sie keine einfachen Einfüge- und Lösch-Operationen von Knoten, sondern abstrahieren davon (Li u. a. 2008; Li u. a. 2010). Diese sogenannten *High-Level Change*

Operations kombinieren einfache Änderungsoperationen zu komplexeren Operationen. Für die Ähnlichkeitsberechnung wird schließlich die Anzahl an Änderungsoperationen gezählt, um ein Prozessmodell in ein anderes zu transformieren. Je weniger solche Operationen notwendig sind, umso ähnlicher sind sich zwei Modelle. Und auch das in (Abbas und Seba 2012) beschriebene Verfahren berechnet eine Graph-Editier Distanz. Das Matching von Knoten beruht dabei auf einem semantischen Ähnlichkeitsmaß für Wörter mit Hilfe von WordNet. Zur Beschleunigung der Berechnung wird zudem die Graphstruktur in einen Baum umgeformt. In (Martens u. a. 2014) wird zur Verbesserung der Laufzeit zur Berechnung der Graph-Editier Distanz ein genetischer Algorithmus entworfen. Das Matching von Knoten wird allerdings nicht vertieft betrachtet, sondern es wird davon ausgegangen, dass identische Aktivitäten gleich beschriftet sind.

Auch in (Wasser und Lincoln 2013) wird eine Graph-Editier Distanz zur Ähnlichkeitsberechnung verwendet. Ein Matching von Knoten wird mit Hilfe speziell erstellter Taxonomien berechnet, die hierarchische Beziehungen zwischen Aktivitäten bzw. Geschäftsobjekten sowie die Ausführungsreihenfolge von Aktivitäten beschreiben. Übereinstimmende Knoten werden durch die Distanz in diesen Taxonomien ermittelt. Schließlich ergänzen Cao u. a. (2016) strukturelle Ansätze zur Ähnlichkeitsberechnung auf Basis einer Graph-Editier Distanz durch eine explizite Berücksichtigung von Stellen. Dies wird in anderen Ansätzen für Petri-Netze nicht berücksichtigt. Darüber hinaus wird das benötigte Matching als Zuordnungsproblem formuliert, sodass sich bei der Evaluation im Vergleich zu anderen Verfahren, die eine Graph-Editier Distanz berechnen, eine kürzere Berechnungszeit ergab.

Bergmann und Gil (2011) und Bergmann und Gil (2014) verwenden hingegen keine Graph-Editier Distanz, sondern bilden die Knoten zweier Prozessmodelle aufeinander ab, sodass die Gesamtähnlichkeit der abgebildeten Knoten maximiert wird. Dabei werden Modelle durch die Verwendung einer Ontologie semantisch annotiert. Diese semantischen Annotationen werden zur Berechnung der Ähnlichkeit von Knoten und Kanten verschiedener Prozessmodelle verwendet. In (Bergmann u. a. 2013) werden zur Clusterung von Prozessmodellen zudem zwei Cluster-Algorithmen eingesetzt, die auf den berechneten Ähnlichkeitswerten operieren. Das in (Fu u. a. 2012) beschriebene Verfahren bildet ebenso die Graphstruktur von Prozessmodellen aufeinander ab. Dazu wird der Graph eines Modells zunächst in eine Baumstruktur transformiert. Zur Berechnung der Ähnlichkeit von zwei Prozessmodellen wird daraufhin die jeweilige Baumstruktur anhand gleich benannter Knoten verglichen. Dieser Ansatz wird auch

in (Eshuis und Grefen 2007) eingesetzt. Dabei wird die Graphstruktur aus Effizienzgründen zunächst in eine Baum-Struktur transformiert, wobei die Ähnlichkeit von zwei Modellen ebenso über die Anzahl sich entsprechender Knoten mit übereinstimmenden Beschriftungen bestimmt wird. Sánchez-Charles u. a. (2016) verwenden auch eine Darstellung von Prozessmodellen als Baumstruktur zur Berechnung eines Ähnlichkeitswerts. Dazu nutzen sie ein Verfahren aus der Phylogenetik[2] und adaptieren dieses für die Ähnlichkeitsberechnung von Prozessmodellen. Dabei gehen sie jedoch nur auf die Struktur ein und setzen ein Matching von Aktivitäten voraus.

Auch in (Madhusudan u. a. 2004) werden die Knoten von Prozessmodellen aufeinander abgebildet. Ein Matching der Knoten wird mit Hilfe von Natural Language Processing und Syntax-basierenden Techniken ermittelt. Dieser Ansatz unterscheidet sich jedoch insbesondere durch die Berechnung der Ähnlichkeit von Knoten von anderen Verfahren. Der für einen Knoten zunächst berechnete Ähnlichkeitswert verändert die Werte angrenzender Knoten, sodass der Ähnlichkeitsgrad eines Knotens durch die Ähnlichkeit der verbundenen Knoten beeinflusst wird. In (Zhuge 2002) werden Modelle dagegen in eine ontologische Graph-Repräsentation überführt, die Spezialisierungsbeziehungen beinhaltet. Aktivitäten zweier Prozessmodelle sind sich dabei ähnlicher, je geringer die Distanz zwischen ihnen in der Ontologie ist und die zugehörigen Modelle sind umso ähnlicher, je ähnlicher die Aktivitäten sind.

In (Uba u. a. 2011) wird zur Erkennung von exakt duplizierten Modellteilen in verschiedenen Modellen auf einen Graph-basierten Ansatz zurückgegriffen, der diese Modellteile durch eine Überprüfung auf Graph-Isomorphie ermittelt. Die eingesetzte Graphstruktur wird letztlich in eine codierte Repräsentation basierend auf den Beschriftungen von Prozessmodellelementen umgeformt. Sollten die Repräsentationen von zwei Modellteilen übereinstimmen, so handelt es sich um eine gesuchte Graph-Isomorphie. Somit würde auch dieser Ansatz ein Matching von Modellelementen voraussetzen, sollten gleiche Elemente in unterschiedlichen Modellen verschiedene Bezeichnungen enthalten. Dieser Ansatz wird in (Ekanayake u. a. 2012; La Rosa u. a. 2013) zur Erkennung sogenannter „Approximate Clones“ erweitert, indem das Verfahren aus (Dijkman u. a. 2009b) zur Berechnung der Ähnlichkeit von Prozessmodellen mit der zuvor beschrieben Identifikationstechnik aus (Uba u. a. 2011) kombiniert wird. Die Ähnlichkeitsberechnung aus (Dijkman u. a. 2009b) wird zusätzlich durch

[2] Eine Fachrichtung der Genetik und Bioinformatik, die sich mit der Erforschung von Abstammungen beschäftigt.

die Einbeziehung einer semantischen Ähnlichkeitsberechnung von Wörtern in Beschriftungen mit Hilfe von WordNet sowie die Berücksichtigung des Kontrollflusses erweitert. Des Weiteren werden in (Dumas u. a. 2013a) Verbesserungsmöglichkeiten hinsichtlich der Laufzeit beschrieben.

Bae u. a. (2006a) verwenden zum einen die Anzahl der übereinstimmenden Knoten zur Bestimmung der Ähnlichkeit von Prozessmodellen, wobei nicht festgelegt wird, wie genau eine solche Übereinstimmung bestimmt wird (Bae u. a. 2006a; Bae u. a. 2006b; Bae u. a. 2007). Zum anderen wird aus dem Kontrollfluss von Prozessmodellen eine Blockstruktur erzeugt, die explizit Konnektoren berücksichtigt. Die Distanz verschiedener Blockstrukturen wird schließlich für diesen strukturellen Ähnlichkeitsvergleich verwendet (Bae u. a. 2006a). Alternativ wird die Struktur von Modellen durch Matrizen repräsentiert, deren Distanz auch die Distanz der Modelle darstellt (Bae u. a. 2006b; Bae u. a. 2007). Gerth u. a. (2010) transformieren die Graphstruktur von Prozessmodellen ebenfalls in eine Blockstruktur ähnlich zu (Bae u. a. 2006a). Bei dem Verfahren von Gerth u. a. (2010) wird allerdings kein Ähnlichkeitswert berechnet, sondern darüber entschieden, ob zwei Modelle oder Teile daraus äquivalent zueinander sind. Das dafür notwendige Matching von Prozessmodellelementen wird über die Gleichheit von Bezeichnungen bzw. die Levenshtein-Distanz ermittelt (Gerth u. a. 2011). Auch Dijkman u. a. (2011) beziehen sich auf die Erkennung ähnlicher Modellteile. In diesem Verfahren werden Matches zwischen zwei Prozessmodellen berechnet und die Ähnlichkeit von Modellteilen über die Anzahl der Matches in diesen Modellteilen bestimmt.

Für die Ähnlichkeitsberechnung in (Gacitua-Decar und Pahl 2009; Gacitua-Decar und Pahl 2010) wird die Graphstuktur von Prozessmodellen aufeinander abgebildet. Dabei werden insbesondere die Beschriftungen von Modellelementen berücksichtigt. So wird mit Hilfe von lexikalischen Korpora die Ähnlichkeit einzelner Wörter bestimmt und es wird die Reihenfolge und Häufigkeit von Wörtern in Beschriftungen berücksichtigt. Auch in (Montani u. a. 2015) wird zur Berechnung einer Graph-Editier Distanz der Fokus auf zusätzliche Daten gelegt, um das zugrunde liegende Matching zu verbessern. Hierbei werden Informationen von Prozessausführungen wie die Häufigkeit von aufeinanderfolgenden Aktivitäten und die Dauer von Aktivitäten berücksichtigt. Zusätzlich wird die semantische Ähnlichkeit von Beschriftungen durch semantische Ähnlichkeitsmaße für Wörter miteinbezogen, um übereinstimmende Knoten zur Berechnung der Graph-Editier Distanz zu finden.

Jung und Bae (2006) nutzen neben dem Vergleich von Aktivitätsbeschriftungen ebenfalls die Graphstruktur von Prozessmodellen zur Berechnung eines Ähnlichkeitswerts (Jung und Bae 2006; Jung u. a. 2009). Dazu werden Prozessmodelle zunächst in einem Vektorraum mit den Aktivitätsbeschriftungen als Dimensionen dargestellt, sodass jedes Modell einem Vektor entspricht. Mit Hilfe der Kosinus-Ähnlichkeit wird dann der Ähnlichkeitsgrad von Modellen berechnet, die anschließend geclustert werden. Innerhalb jedes Clusters werden die darin enthaltenen Modelle noch einmal geclustert. Hierzu werden die Modelle wiederum als Vektoren dargestellt, wozu die die Aktivitäten verbindenden Kanten und somit die Graphstruktur herangezogen werden. Die Ähnlichkeit wird daraufhin wieder mit Hilfe der Kosinus-Ähnlichkeit bestimmt. Dieser Ansatz wird in (Kastner u. a. 2009) um eine Möglichkeit erweitert, hierarchische Prozessmodelle, die Subprozesse enthalten, vergleichen zu können.

Als weitere Optionen zur Ähnlichkeitsberechnung erläutern sowohl (Lam 2009) als auch (Huang u. a. 2004) jeweils einen Ansatz bezüglich der Graphstruktur Dimension, bei dem Modelle in spezifische Arten von Graphen zur Ähnlichkeitsberechnung transformiert werden. Das Verfahren in (Lam 2009) zielt allerdings nur auf die Äquivalenzprüfung von Prozessmodellen ab. Zwei Modelle sind dabei äquivalent, wenn die jeweiligen Graphen identisch sind. (Huang u. a. 2004) fokussiert sich hingegen im Kontext der Suche nach Webservices vor allem auf eine Ähnlichkeitsberechnung zwischen Service-Beschreibungen, die zu spezifisch auf den Anwendungskontext hin entworfen wurde, um sie für eine allgemeine Suche nach Prozessmodellen einsetzen zu können.

Verhalten: Die Bestimmung äquivalenter Modelle ist ebenso das Ziel von (Hidders u. a. 2005). In diesem Ansatz wird die Verhaltensdimension berücksichtigt, indem zwei Modelle als äquivalent aufgefasst werden, wenn ihre Mengen an möglichen Traces übereinstimmen. Van der Aalst u. a. (2006) berechnen demgegenüber einen Ähnlichkeitswert mit Hilfe eines Logs tatsächlicher Prozessausführungen (van der Aalst u. a. 2006; de Medeiros u. a. 2008). Zwei Modelle sind sich demnach umso ähnlicher je mehr Traces aus dem Log möglichst vollständig von beiden Modellen ausgeführt werden können.

Einige weitere frühe Arbeiten zur Berechnung der Ähnlichkeit von Prozessmodellen wurden von Wombacher u. a. (2004) für die Suche nach Services entworfen. Prozesse werden dabei als deterministische endliche Zustandsautomaten beschrieben,

wobei die Ähnlichkeit dieser Modelle mit Hilfe der aus den Zustandsautomaten erzeugten Sprachen bestimmt wird. Zwei Prozesse sind kompatibel bzw. ähnlich, wenn der Durchschnitt der Sprachen nicht leer ist, ansonsten sind sie nicht kompatibel. Somit wird das Verhalten der Prozesse sowie die resultierenden Sprachen syntaktisch verglichen. Weitere Varianten syntaktischer Vergleiche wurden daraufhin in (Wombacher und Rozie 2006; Wombacher 2006) untersucht, während die Einbeziehung der Struktur von Prozessmodellen in (Wombacher und Li 2010) analysiert wurde. Auch das Verfahren in (Huang u. a. 2009) zur Empfehlung von Services verwendet endliche Zustandsautomaten und vergleicht das Verhalten von Modellen.

Mendling u. a. (2007b) fokussieren sich bei der Berechnung der Ähnlichkeit von Prozessmodellen auch auf die Verhaltensperspektive (Mendling u. a. 2007b; van Dongen u. a. 2008). Dazu setzen sie sogenannte *Causal Footprints* ein, die für Aktivitäten eines Modells die jeweils zuvor und anschließend durchgeführten Aktivitäten beinhalten. Daraus lässt sich eine Reihenfolgenbeziehung zwischen den Aktivitäten eines Prozessmodells erstellen. Die Causal Footprints werden als Vektoren repräsentiert und die Ähnlichkeit von Modellen über die Kosinus-Ähnlichkeit berechnet. Zur Abbildung der Aktivitäten wird auf drei Varianten eines Prozessmodell-Matching Ansatzes zurückgegriffen. Bei der ersten Variante wird verglichen, wie viele Wörter in den Aktivitätsbeschriftungen übereinstimmen bzw. Synonyme zueinander sind. Die zweite bezieht die vor und nach einer Aktivität auftretenden Ereignisse mit ein und die dritte nutzt eine syntaktische Editier-Distanz zwischen Beschriftungen.

Esgin und Senkul (2011) beschreiben einen zu den *Causal Footprints* verwandten Ansatz, bei dem sowohl die Verhaltens- als auch die Graphstruktur Dimension verwendet werden. Sie ermitteln für jede Aktivität zweier Prozessmodelle alle Vorgänger- und Nachfolgeraktivitäten. Der größte gemeinsame Teilgraph jedes Paares von Aktivitäten ergibt daraufhin ein Element eines Gewichtsvektors. Die auf das Intervall $[0, 1]$ normierte euklidische Norm stellt schließlich den Ähnlichkeitswert zwischen zwei Prozessmodellen dar. In (Esgin und Karagoz 2013b; Esgin und Karagoz 2013a) wird dieser Ansatz durch den Einsatz eines genetischen Algorithmus ergänzt, um die Laufzeit zu verbessern.

Auch Kunze u. a. (2011b) betrachten vorrangig die Verhaltensdimension von Prozessmodellen für eine ähnlichkeitsbasierte Suche (Kunze u. a. 2011b; Kunze u. a. 2011a). Dazu verwenden sie *Behavioral Profiles* (Weidlich u. a. 2011), die die Reihenfolgenbeziehungen von Aktivitäten in Prozessmodellen repräsentieren. Zwei Prozessmo-

delle sind dabei umso ähnlicher, je mehr sich ihre Behavioral Profiles entsprechen. Dieser Ansatz basiert ebenfalls auf einem Prozessmodell-Matching, wobei darauf in (Kunze u. a. 2011a) nicht gesondert eingegangen, sondern von identisch bezeichneten, sich entsprechenden Modellelementen ausgegangen wird. In (Kunze und Weske 2012; Kunze u. a. 2013) wird der Fokus etwas abgeändert, sodass nun ausgehend von einem Anfragemodell Modelle gesucht werden, die die Reihenfolgenbeziehungen der Aktivitäten des Anfragemodells wiedergeben können. Dies kann z. B. bei der Suche nach Modellen helfen, wenn in dem Anfragemodell nur einige wesentliche Aktivitäten modelliert sind. Baumann u. a. (2015a) beschreiben einen Ansatz, der zudem das Verhalten von Prozessmodellen durch die Betrachtung von optionalen und sich wiederholenden Aktivitäten berücksichtigt. Dabei wird die Strukturdimension anhand der Länge von kürzesten Pfaden mit in die Ähnlichkeitsberechnung einbezogen.

Becker u. a. (2011) kombinieren alle möglichen Ausführungssequenzen von Aktivitäten (Traces) mit festgelegten Ausführungswahrscheinlichkeiten. Daraus wird für ein Prozessmodell eine Wahrscheinlichkeitsverteilung bestimmt. Zwei Prozessmodelle werden schließlich anhand ihrer Wahrscheinlichkeitsverteilungen miteinander verglichen, wobei grundlegend von einem gegebenen Matching von Aktivitäten ausgegangen wird.

Das Verfahren von Wang u. a. (2010) zielt auf den Verhaltensaspekt von Prozessmodellen ab. Für beschriftete Petri-Netze wird ein Ähnlichkeitswert basierend auf der Übereinstimmung von Schaltfolgen bestimmt. In (Wang u. a. 2012) wird darüber hinaus die Graphstruktur berücksichtigt, indem zwei Knoten anhand der Anzahl an Vorgängern und Nachfolgern sowie der Beschriftungen verglichen werden. Dies wird in ähnlicher Weise auch in dem in (Belhoul u. a. 2012) beschriebenen Verfahren umgesetzt. In diesem Ansatz wird allerdings nicht die Übereinstimmung von Schaltfolgen, sondern die Gesamtdistanz aller Schaltfolgen ermittelt. In (Belhoul u. a. 2015) wird das Verfahren durch die Berücksichtigung der Struktur von Prozessmodellen erweitert. In (Sun 2010) wird für Schaltfolgen dagegen nicht die Gesamtdistanz berechnet, sondern die Ähnlichkeit wird anhand einer Editier-Distanz von Schaltfolgen bestimmt.

Zha u. a. (2009) verwenden zum einen ein Ähnlichkeitsmaß basierend auf der Übereinstimmung der Menge aller möglichen Folgen von Aktivitäten in Prozessmodellen (Zha u. a. 2009; Zha u. a. 2010). Zum anderen werden die Mengen aufeinander folgender Paare von Aktivitäten verglichen, falls alle vollständigen Folgen von Aktivitäten nicht berechenbar sein sollten wie z. B. im Fall von Schleifen in Prozessmodellen.

Implizit wird bei diesem Ansatz von identisch bezeichneten Aktivitäten bzw. dem Vorhandensein von Matches zur Bestimmung eines Ähnlichkeitswerts ausgegangen. In (Jin u. a. 2012) wird zu diesem Verfahren explizit eine Berechnungsmethode für Matches angegeben. Für das Matching werden die in (Jin u. a. 2011) beschriebenen Techniken verwendet. Zudem werden in (Sarno u. a. 2013) die Verfahren aus (Wang u. a. 2010) und (Bae u. a. 2006b) zur Berechnung von Ähnlichkeitswerten miteinander kombiniert. Zudem wird in (Gerke u. a. 2009) die Länge der längsten gemeinsamen Subsequenzen von Schaltfolgen zweier Modelle zur Berechnung eines Ähnlichkeitsgrads eingesetzt.

Des Weiteren wird in (Lu und Sadiq 2007; Lu u. a. 2009) ein Ansatz zur Ähnlichkeitsberechnung speziell für Prozessvarianten beschrieben. Varianten sind Ausführungen von Prozessen, die auf dem gleichen Modell basieren, allerdings Unterschiede wie z. B. eine abweichende Ausführungsreihenfolge von Aktivitäten aufweisen. Das beschriebene Ähnlichkeitsmaß ist auf die Suche nach solchen Varianten ausgelegt. Eine Anfrage bietet dabei die Möglichkeit zur Spezifizierung spezieller Bedingungen wie etwa einer gewissen Ausführungsreihenfolge. Varianten sind schließlich umso ähnlicher zu einer Anfrage, je mehr Bedingungen erfüllt werden. Auf die Ähnlichkeitsberechnung selbst wird in (Lu und Sadiq 2007; Lu u. a. 2009) nur kurz eingegangen. Im Wesentlichen werden mögliche Traces und die Abbildung der Graphen von Prozessmodellen aufeinander berücksichtigt. Diese Ähnlichkeitsberechung wird für die strukturelle Dimension in (Mahmod und Chiew 2010; Mahmod und Radzi 2010) durch eine Graph-Editier Distanz ergänzt. Schließlich fügt (Liu u. a. 2012) für die verhaltenszentrierten Ähnlichkeitsmaße noch die Berücksichtigung von Datenobjekten zur Berechnung eines Ähnlichkeitswerts hinzu.

Ein Ansatz, der neben der Verhaltensdimension auch die Beschriftungen von Modellelementen berücksichtigt, ist in (Nejati u. a. 2007) beschrieben. Bezüglich der Verhaltensdimension wird die Ähnlichkeit von zwei Modellen per Bisimulation bestimmt. Hinsichtlich der Beschriftungen sind zwei Modelle ähnlich, wenn sie aus möglichst vielen gleichen N-Grammen bestehen oder indem eine durch das WordNet Paket (Pedersen u. a. 2004) bestimmte semantische Ähnlichkeit gegeben ist.

Natürliche Sprache: Weitere frühe Arbeiten von Castano und Antonellis (1995) zielen bereits auf textuelle Beschreibungen von Prozessen ab (Castano und Antonellis 1995; Castano und Antonellis 1996). Bei diesem Ansatz wird auf keine der heutzutage

vorherrschenden Modellierungssprachen für Prozesse, sondern auf eine eigene Darstellungsweise als Tupel zurückgegriffen. Solche Tupel enthalten z. B. Mengen von Termen zur Beschreibung von Aktivitäten, Daten und Geschäftsobjekten. Die Ähnlichkeitsberechnung erfolgt zum einen über einen syntaktischen Identitätsvergleich der Terme. Zum anderen wird ein für jeden Anwendungsfall manuell erstelltes Thesaurus eingesetzt, um die Ähnlichkeit von Termen im Falle von Synonymen und Homonymen zu quantifizieren.

Ein sehr einfacher Ansatz, der auf natürlichsprachliche Beschriftungen fokussiert, ist in (Akkiraju und Ivan 2010) beschrieben, bei dem lediglich die Anzahl an gleich benannten Aktivitäten zur Bestimmung eines Ähnlichkeitswerts verwendet wird. Gao u. a. (2007) messen die Ähnlichkeit von Prozessmodellen, indem sie ebenfalls die Beschriftungen von Aktivitäten miteinander vergleichen (Gao u. a. 2007; Gao und Zhang 2009). Dabei wird neben einem syntaktischen Vergleich durch eine Editier-Distanz auch ein semantischer Ähnlichkeitvergleich mit Hilfe einer Ontologie durchgeführt. Des Weiteren werden in (Gao und Zhang 2009) die Beschriftungen von Datenobjekten und Ressourcen sowie der Datenfluss berücksichtigt. Ein weiterer einfacher Ansatz ist in (Srivastava und Mukherjee 2009) beschrieben. In diesem Ansatz werden für Aktivitäten und Prozessmodelle bestimmte Eigenschaften wie die Bezeichnung eines Modells definiert und diese syntaktisch miteinander verglichen. Je ähnlicher sich diese Eigenschaften sind, umso ähnlicher sind sich zwei Modelle.

In (Ehrig u. a. 2007; Brockmans u. a. 2006; Koschmider und Oberweis 2007) werden Prozessmodelle in ein Ontologie-Datenformat transformiert. Die Ähnlichkeit der Modelle wird dann durch drei Perspektiven ausgedrückt. Es wird die syntaktische Ähnlichkeit von Elementbezeichnungen durch die Levenshtein-Distanz quantifiziert und die semantische Ähnlichkeit von Beschriftungen durch das Auffinden von Synonymen mit Hilfe von WordNet bestimmt. Darüber hinaus wird die Graphstruktur von Modellen berücksichtigt, indem die Ähnlichkeit von in der Ontologie-Darstellung verbunden Knoten miteinbezogen wird. Auch in (Wang u. a. 2007) werden Modelle anhand der Distanz von Aktivitäten in spezifisch für die Ähnlichkeitsberechnung erstellten Ontologien verglichen.

Fengel und Reinking (2012) verwenden einen vergleichbaren Ansatz, der neben einer Ontologie-Transformation von Modellen auch Techniken aus dem Natural Language Processing zur Bestimmung der Ähnlichkeit von Prozessmodellen nutzt (Fengel und

Reinking 2012; Humm und Fengel 2012; Fengel 2014). Als Basis der Ähnlichkeitsberechnung dienen Matches zwischen den Elementbeschriftungen der Modelle. Zur Berechnung von Matches werden NLP-Techniken wie die Wortstammreduktion oder die Entfernung von Stoppwörtern eingesetzt. Die Ähnlichkeit der resultierenden Beschriftungen wird neben einem syntaktischen Vergleich auch durch das Auffinden von Synonymen durch WordNet oder Ontology Matching Verfahren semantisch bestimmt. Zudem wird die Stuktur-Dimension über einen Vergleich der Anzahl an Schleifen in den Modellgraphen in die Ähnlichkeitsberechnung miteinbezogen.

In (Pittke u. a. 2012) wird das auf die Verhaltensdimension fokussierende Verfahren aus (Kunze u. a. 2011a) um einen semantischen Vergleich von Beschriftungen ergänzt. Dazu werden Aktivitätsbeschriftungen in die Teile Aktion, Geschäftsobjekt und zusätzliche Informationen zerlegt. Die Ähnlichkeit dieser Teile wird für zwei Beschriftungen unterschiedlicher Modelle jeweils durch ein semantisches Ähnlichkeitsmaß für Wörter, das auf WordNet zurückgreift, bestimmt. Für die Ähnlichkeitsberechnung zweier Modelle muss dann sowohl die Gesamtähnlichkeit aller Beschriftungen als auch die verhaltensbasierte Ähnlichkeit über einem Schwellenwert liegen.

Auch Niemann u. a. (2010) fokussieren sich in ihrem Ansatz auf die Beschriftungen von Modellelementen (Niemann u. a. 2010; Niemann u. a. 2012). Neben einer syntaktischen Ähnlichkeitsberechnung von Beschriftungen mit der Levenshtein-Distanz, wird die semantische Ähnlichkeit berücksichtigt. Dazu werden mit Hilfe von WordNet Synonyme ermittelt und die einem Wort vorausgehenden und nachfolgenden Wörter miteinbezogen. Diese Ähnlichkeitsberechnungen von Beschriftungen werden zur Ermittlung von Matches durchgeführt, auf dessen Basis unter Berücksichtigung der Graphstruktur Cluster ähnlicher Modellelemente in zwei Prozessmodellen ermittelt werden. Die Gesamtähnlichkeit eines Modells wird dann letztlich durch die Ähnlichkeit dieser Cluster sowie möglicher weiterer Modellelemente, die in keinem Cluster vorhanden sind, bestimmt.

Das in (Tka und Ghannouchi 2012) beschriebene Verfahren ist zu dem Vorangegangenen sehr ähnlich. Auch in diesem Ansatz werden hauptsächlich die Beschriftungen miteinander verglichen, um Matches zu ermitteln. Es wird auf syntaktischer Ebene die Levenshtein-Distanz unter Berücksichtigung der Wortreihenfolge berechnet und auf semantischer Ebene werden Synonyme miteinbezogen. Zusätzlich wird die Graphstruktur hinzugezogen, indem überprüft wird, wie viele Kanten zwischen zwei Modellen basierend auf den gefundenen Matches übereinstimmen. Auch das in (Awad

u. a. 2008) vorgestellte Verfahren setzt bei den Aktivitätsbeschriftungen an. Es wird eine spezielle Ontologie erstellt, mit deren Hilfe Synonyme ermittelt werden, sodass diese beim Matching von Aktivitäten berücksichtigt werden können.

Malinova u. a. (2013) verfolgen einen grundlegend eng verwandten Ansatz zu der im nächsten Kapitel vorgestellten LS3-Suche. Sie bilden ebenso wie der LS3-Ansatz Prozessmodelle in einen Vektorraum ab, verwenden zur Auffindung ähnlicher Modelle allerdings kein Ähnlichkeitsmaß zwischen Vektoren, sondern ein klassisches Clustering-Verfahren, da sie als Ziel die Clusterbildung von Prozessmodellen verfolgen. Somit kann dieses Verfahren nicht direkt zur Suche nach zu einem Anfragemodell ähnlichen Modellen verwendet werden. Des Weiteren verwendet dieser Ansatz einen syntaktischen Vergleich von Termhäufigkeiten, während das LS3-Verfahren die Semantik von textuellen Beschriftungen durch die LSA miteinbezieht.

Qiao u. a. (2011) beschreiben den zu dem LS3-Verfahren ähnlichsten Ansatz. Dieser nutzt eine Technik zur Sprachenmodellierung, die Latent Dirichlet Allocation (LDA), um Themen eines Modells zu identifizieren. Zusätzlich wird ein Graph-struktureller Vergleich sowie ein Clustering von Modellen eingesetzt. Während die im Folgenden verwendete LSA Vektoren in einem k-dimensionalen Raum zur Bestimmung eines Ähnlichkeitswerts einsetzt, beruht der LDA-Ansatz auf der Bestimmung der Wahrscheinlichkeit, dass ein Prozessmodell ein Anfragemodell erzeugen kann. Auf dieser textuellen Analyse aufbauend werden Cluster ähnlicher Modelle berechnet, die erneut anhand ihrer Graphstruktur in Cluster ähnlicher Modelle unterteilt werden.

Sonstiges: Schließlich existieren noch drei Ansätze zur Messung der Ähnlichkeit von Prozessmodellen, die ihren Fokus nicht auf eine der zuvor genannten Kategorien legen. Der erste Ansatz verwendet Prozessmodell-Metriken, die bestimmte Eigenschaften eines Modells wie etwa die Anzahl an Startknoten oder die Kontrollflusskomplexität repräsentieren (Melcher und Seese 2008). Anschließend werden die Modelle anhand der Metrikwerte durch einen Cluster-Algorithmus in Gruppen ähnlicher Modelle eingeteilt. Der zweite Ansatz bestimmt die Ähnlichkeit von Modellen anhand der Anzahl übereinstimmender Schlagworte (Laue und Becker 2012). Dabei werden Schlagworte von Nutzern vergeben, womit dieser Ansatz das einzig identifizierte Ähnlichkeitsmaß ist, das anstatt Modellinformationen die Eingabe von Nutzern verwendet. Das dritte Verfahren verwendet schließlich Compliance Regeln zur Bestimmung eines Ähnlichkeitswerts (Rinderle-Ma und Kabicher-Fuchs 2016). Zwei

Modelle sind sich dabei umso ähnlicher, je mehr Compliance Regeln für beide Modelle gelten im Verhältnis zu allen Regeln, die für die beiden Modelle zutreffen.

Ein zusammenfassender Überblick der verwandten Arbeiten ist in den Tabellen 5.1 und 5.2 dargestellt. Darin repräsentieren die Zeilen jeweils unterschiedliche Verfahren zur Berechnung eines Ähnlichkeitswerts für Prozessmodelle. Teilweise sind mehrere Referenzen angegeben, wenn das grundlegende Verfahren in weiteren Arbeiten ergänzt wurde. Die markierten Spalten geben jeweils an, welche Dimensionen ein Ansatz berücksichtigt. Schließlich ist in der letzten Spalte markiert, ob ein Prozessmodell-Matching zur Berechnung eines Ähnlichkeitswerts notwendig ist. Aus den Tabellen ist ersichtlich, dass die meisten Verfahren zur Ähnlichkeitsberechnung ein Matching von Modellelementen verwenden. Wie in Kapitel 4 beschrieben, ist dies jedoch weiterhin eine große Herausforderung, was auch in den Wettbewerben zum Prozessmodell-Matching erkannt wurde (Cayoglu u. a. 2014; Antunes u. a. 2015). Die im Folgenden erläuterten LS3-Suchverfahren benötigen demgegenüber kein Matching von Prozesselementen und umgehen damit diese Schwierigkeit.

Des Weiteren nutzen die aufgelisteten Verfahren häufig Techniken, die zu den Dimensionen Graphstuktur bzw. Verhalten von Prozessmodellen zählen, um einen Ähnlichkeitsgrad zu bestimmen. Die Beschriftungen von Modellelementen werden dabei typischerweise lediglich für das zugrunde liegende Matching berücksichtigt. Häufig dafür eingesetzte Mittel sind die Bestimmung von Matches über eine Editier-Distanz von Beschriftungen sowie der Einsatz von Ontologien und Thesauri wie WordNet, um die Verwendung von Synonymen zu erkennen. Die Dimension der natürlichen Sprache steht vor allem in den Arbeiten von Malinova u. a. (2013) und Qiao u. a. (2011) im Vordergrund. Auf diese Dimension beschränken sich auch die LS3-Verfahren, die sich von (Malinova u. a. 2013) insbesondere durch die Berücksichtigung der Semantik von textuellen Beschriftungen unterscheiden. Im Gegensatz zu (Qiao u. a. 2011), das ebenfalls die Semantik einbezieht, wird auf einen strukturellen Vergleich von Modellen verzichtet, wodurch ein Matching von Modellelementen entfällt.

5.3 LS3: Latent Semantic Analysis-based Similarity Search

In diesem Abschnitt wird die *Latent Semantic Analysis-based Similarity Search* (LS3) beschrieben. Dazu wird im Folgenden zunächst ausgeführt, wie sich für alle Modelle

Tab. 5.1: Übersicht von Ansätzen zur Berechnung der Ähnlichkeit von Prozessmodellen

Quellen	Natürliche Sprache		Graphstruktur		Verhalten	Nutzer	Matching
	Syntax	Semantik	Graph	Kontrollfluss			
Abbas und Seba (2012)		x		x			x
Akkiraju und Ivan (2010)	x						x
Awad u. a. (2008)	x	x					x
Bae u. a. (2006a), Bae u. a. (2006b) und Bae u. a. (2007)				x			x
Becker u. a. (2011)	x				x		x
Belhoul u. a. (2012) und Belhoul u. a. (2015)	x			x	x		x
Bergmann und Gil (2011), Bergmann u. a. (2013) und Bergmann und Gil (2014)	x			x			x
Cao u. a. (2016)	x		x				x
Castano und Antonellis (1995) und Castano und Antonellis (1996)	x	x					
Dijkman u. a. (2009b), Kunze und Weske (2010), Yan u. a. (2010), Yan u. a. (2012c), Gao u. a. (2013), Ling u. a. (2014) und Liu u. a. (2014)	x		x				x
Dijkman u. a. (2011)	x			x			x
Ehrig u. a. (2007), Brockmans u. a. (2006) und Koschmider und Oberweis (2007)	x	x	x				x
Ekanayake u. a. (2012), La Rosa u. a. (2013) und Dumas u. a. (2013b)	x	x	x	x			x
Esgin und Senkul (2011), Esgin und Karagoz (2013b) und Esgin und Karagoz (2013a)			x		x		x
Eshuis und Grefen (2007)	x		x	x			x
Fengel und Reinking (2012), Humm und Fengel (2012) und Fengel (2014)	x	x	x				x
Fu u. a. (2012)	x		x	x			x
Gacitua-Decar und Pahl (2009) und Gacitua-Decar und Pahl (2010)		x	x	x			x
Gao u. a. (2007) und Gao und Zhang (2009)	x	x					x
Gerke u. a. (2009)					x		x
Gerth u. a. (2010) und Gerth u. a. (2011)	x			x			x
Grigori u. a. (2006), Corrales u. a. (2006), Grigori u. a. (2008), Grigori u. a. (2010), Gater u. a. (2010), Gater u. a. (2011) und Gater u. a. (2012)	x	x		x			x
Hidders u. a. (2005)					x		x
Huang u. a. (2009)					x		x
Huang u. a. (2004)				x			x

Tab. 5.2: Fortsetzung der Übersicht von Ansätzen zur Berechnung der Ähnlichkeit von Prozessmodellen

Quellen	Natürliche Sprache		Graphstruktur		Verhalten	Nutzer	Matching
	Syntax	Semantik	Graph	Kontrollfluss			
Jung und Bae (2006), Jung u. a. (2009) und Kastner u. a. (2009)	x		x	x			x
Kunze u. a. (2011b), Kunze u. a. (2011a), Kunze und Weske (2012) und Kunze u. a. (2013)	x				x		x
Laue und Becker (2012)						x	
Li u. a. (2008) und Li u. a. (2010)			x		x		x
Liu u. a. (2012)	x				x		x
Lu und Sadiq (2007), Lu u. a. (2009), Mahmod und Chiew (2010) und Mahmod und Radzi (2010)			x	x			x
Madhusudan u. a. (2004)	x	x		x			x
Malinova u. a. (2013)	x						
Martens u. a. (2014)	x			x			x
Melcher und Seese (2008)				x			
van Dongen u. a. (2008)	x	x			x		x
Minor u. a. (2007)			x	x			x
Montani u. a. (2015)	x	x	x	x	x		x
Nejati u. a. (2007)	x	x			x		x
Niemann u. a. (2010) und Niemann u. a. (2012)	x	x		x			x
Pittke u. a. (2012)		x			x		x
Qiao u. a. (2011)	x	x	x				x
Sánchez-Charles u. a. (2016)	x			x			x
Srivastava und Mukherjee (2009)	x				x		
Sun (2010)	x				x		
Tka und Ghannouchi (2012)	x	x	x				x
Uba u. a. (2011)			x	x			
van der Aalst u. a. (2006) und de Medeiros u. a. (2008)			x		x		x
Wang u. a. (2010), Wang u. a. (2012) und Sarno u. a. (2013)					x		x
Wang u. a. (2007)		x					x
Wasser und Lincoln (2013)	x	x	x		x		x
Wombacher u. a. (2004), Wombacher und Rozie (2006), Wombacher (2006) und Wombacher und Li (2010)	x				x		x
Zha u. a. (2009), Zha u. a. (2010) und Jin u. a. (2012)	x	x			x		x
Zhuge (2002)	x		x				x

einer Prozessmodellbibliothek ähnliche Modelle ermitteln lassen, die einen gewissen Mindestgrad an Ähnlichkeit aufweisen. Dieser Ansatz kann beispielsweise dazu verwendet werden, Duplikate in einer Modellbibliothek zu ermitteln. Um das Verständnis des Verfahrens zu erleichtern, wird im folgenden Abschnitt ein Beispiel zur Ergebnisberechnung ausgeführt. Anschließend wird in Kapitel 5.3.2 erläutert, wie sich ein spezifisches Prozessmodell als Anfrage an eine Modellbibliothek verwenden lässt. Dazu wird jeweils die in Kapitel 3.3 eingeführte Latent Semantic Analysis verwendet. Schließlich wird in Abschnitt 5.3.3 beschrieben, wie das Einfügen und Löschen von Modellen sowie Änderungen existierender Modelle gehandhabt werden können.

Eine ähnlichkeitsbasierte Suche zwischen allen Prozessmodellen einer Prozessmodellbibliothek (LS3-QueryAll Ansatz) wird nach den im Folgenden erläuterten fünf Schritten durchgeführt (siehe Abbildung 5.1 zur Übersicht). Dabei bestehen die ersten vier Schritte aus der Ähnlichkeitsberechnung des *Latent Semantic Analysis-based Similarity Measure* (LSSM), während im fünften Schritt die letztendliche Ergebnisberechnung durchführt wird.

1. **Extraktion von Termen zur Erstellung einer Term-Dokument Matrix:**
   Zur Erstellung einer Term-Dokument Matrix muss jedes Prozessmodell als ein Dokumentvektor d_m entsprechend Definition 5.1 repräsentiert werden. Diese Vektoren stellen dann die Spalten der Term-Dokument Matrix nach Schritt 1 in Abbildung 5.1 dar. Für die in dieser Arbeit verwendeten Petri-Netz basierten Modelle werden alle Terme aus den Stellen- und Transitionsbeschriftungen aller Modelle in $\mathbf{M}$ extrahiert (Menge $\mathfrak{L}$ aus Definition 2.7).

   Anschließend werden drei Vorverarbeitungsschritte ausgeführt: (1) Alle Wörter werden klein geschrieben, (2) Stoppwörter werden entfernt und (3) die verbliebenen Wörter werden mit Hilfe des qualitativ hochwertigen und verbreiteten (Jivani 2011) Stemming-Verfahrens nach Porter (Porter 1980) auf ihre Wortstämme reduziert. Diese Vorverarbeitung wird durchgeführt, um den resultierenden Vektorraum zu verkleinern, indem Terme, die nichts zur Semantik eines Modells beitragen (Stoppwörter), entfernt werden. Durch das Stemming sollen zudem Wörter, die den gleichen Wortstamm besitzen einem Term zugeordnet werden und dessen Anzahl erhöhen, anstatt als separate Terme in der Term-Dokument Matrix aufgelistet zu werden (siehe auch die Kapitel 3.1 und 3.3 für weitere Informationen zu den Vorverarbeitungsschritten).

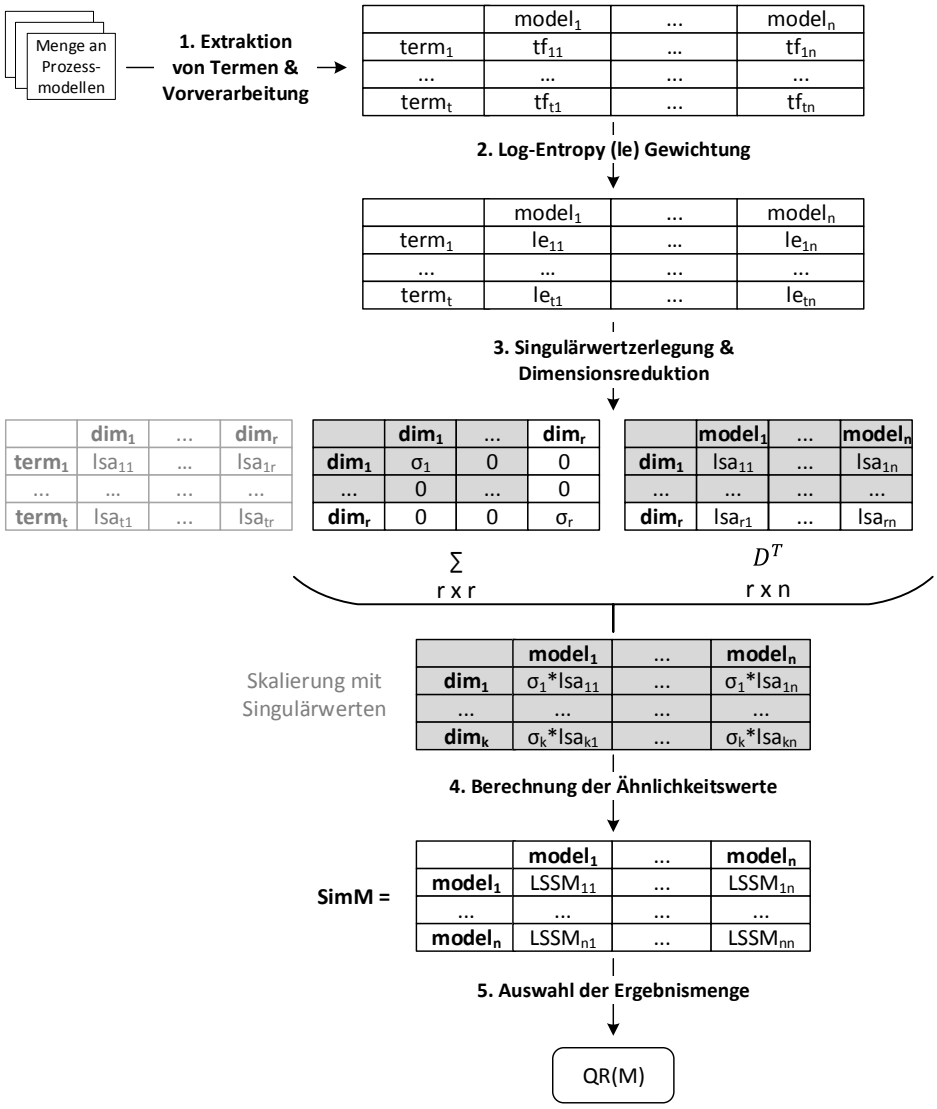


Abb. 5.1: Konzeptueller LS3-Suchansatz zur Berechnung aller ähnlichen Modelle in einer Prozessmodellbibliothek (in Anlehnung an (Schoknecht und Oberweis 2017))

Am Ende dieses Schritts entsteht schließlich eine Term-Dokument Matrix, die absolute Termhäufigkeiten t_{ij} als Einträge enthält. Dabei gilt $1 \leq i \leq t$ mit t entsprechend der Anzahl einzigartiger Terme und $1 \leq j \leq n$ mit n entsprechend

der Anzahl an Modellen in einer Prozessmodellbibliothek **M**. Diese Termhäufigkeiten geben an, wie oft ein Term i in einem Modell j vorkommt. Dieser Schritt wird durch Zeile 1 in Algorithmus 5.1 berechnet.

2. **Transformation der Term-Dokument Matrix:**
   Da die Nutzung absoluter Termhäufigkeiten, wie sie bislang in der Term-Dokument Matrix nach Schritt 1 vorhanden sind, in verschiedenen Untersuchungen nicht empfohlen wird (Dumais 1991; Zaman und Brown 2010), werden die Termhäufigkeiten durch eine Gewichtungsfunktion angepasst. Hierzu wird die *Log-Entropie* (Landauer u. a. 1998) entsprechend folgender Formel eingesetzt:

$$le_{ij} = log_2(tf_{ij} + 1) \cdot (1 + \sum_{j=1}^{n} \frac{p_{ij} \cdot log_2 p_{ij}}{\log_2 n}) \ , \ \forall\, tf_{ij} > 0. \tag{5.2}$$

   Dabei ist $p_{ij} = \frac{tf_{ij}}{gf_i}$ der Quotient aus der absoluten Termhäufigkeit tf_{ij} und der globalen Häufigkeit gf_i. Die globale Häufigkeit gibt an, wie oft ein Term i in der gesamten Prozessmodellbibliothek vorkommt. Diese Form der Gewichtung hatte sich bei einem Vergleich verschiedener Gewichtungsfunktionen als vorteilhaft erwiesen (Nakov u. a. 2001).

   Die Log-Entropie Formel besteht dabei aus zwei Bestandteilen. Zum einen wird der Einfluss großer Unterschiede zwischen Termhäufigkeiten durch eine lokale Gewichtungsfunktion mit Hilfe einer logarithmischen Transformation begrenzt. Zum anderen wird als globale Gewichtungsfunktion der Informationsgehalt entsprechend der Shannon-Entropie (Shannon 1948) gewählt, um den Termen ein hohes Gewicht zuzuweisen, die über einen hohen Informationsgehalt verfügen und somit gut zur Unterscheidung von Modellen geeignet sind. Entsprechend erhalten Terme mit niedrigem Informationsgehalt ein geringes Gewicht. Terme besitzen einen hohen Informationsgehalt, wenn sie konzentriert in wenigen Modellen vorkommen. Ein niedriger Informationsgehalt liegt hingegen vor, wenn sie in vielen Modellen auftreten (Cover und Thomas 2006). Siehe dazu auch den zweiten Schritt in Abbildung 5.1, der in Algorithmus 5.1 durch Zeile 2 ausgeführt wird.

3. **Singulärwertzerlegung und Dimensionsreduktion:**
   In diesem Schritt wird die zuvor transformierte Term-Dokument Matrix zunächst durch eine Singulärwertzerlegung in drei Matrizen aufgeteilt (siehe 3.

Algorithmus 5.1 : LS3-QueryAll(**M**,t,k) nach (Schoknecht und Oberweis 2017)

input : Model collection **M**, threshold t, dimensionality parameter k
output : A set of tuples $QR = \{(m, \mathbf{M}_{sim})\}$ each containing a model $m \in \mathbf{M}$ and the set of similar models M_{sim}

```
   /* Calculate similarity value matrix simM */
1  tdm = calculateTDMatrix(M);
2  wtdm = weightTDMatrix(tdm);
3  svd = calculateSVD(wtdm);
4  svdk = calculateReducedSVD (svd,k);
5  simM = calculateLSSM(svdk);
   /* Calculate QueryAll results */
6  QR = ∅;
7  for each (row i in simM) do
8      results = ∅;
9      for each (column j in simM) do
10         if (SimM[i][j] ≥ θ) then
11             Add model m_j to results;
12         end
13     end
14     Add tuple (m_i, results) to QR;
15 end
16 return QR;
```

Schritt in Abbildung 5.1). Für die Bestimmung der Ähnlichkeit von Modellen sind lediglich die Matrizen Σ und D^T von Relevanz.

Die Matrix D^T beinhaltet dabei die berechneten Prozessmodellvektoren, während die Matrix Σ die nach absteigender Größe sortierten Singulärwerte σ_i enthält. Die Anzahl an Singulärwerten größer 0 entspricht dem Rang r der ursprünglichen Term-Dokument Matrix. Somit entspricht dieser Wert einer oberen Grenze zur Auswahl einer Dimensionsanzahl k für die folgende Dimensionsreduktion. Sollte dabei r als Dimensionsanzahl k gewählt werden, sind die im 4. Schritt berechneten Ähnlichkeitswerte gleich denen, die mit der ursprünglichen (gewichteten) Term-Dokument Matrix ermittelt werden könnten. Für Werte von $k < r$ werden die höchsten k Singulärwerte beibehalten, während die restlichen durch „0-Setzen“ für die Ähnlichkeitsberechnung entfernt werden.

Diese Dimensionsreduktion ist in Abbildung 5.1 durch die grau hinterlegten Bereiche der Matrizen angedeutet. Die Festlegung einer Dimensionszahl k ist dabei vom Anwendungsfall abhängig und es gibt keinen generell optimalen Wert im Sinne einer guten Anfragequalität (Dumais 1991). Die Zeilen 3–4 in Algorithmus 5.1 repräsentieren diesen Schritt.

4. **Berechnung der Ähnlichkeitswerte:**
   Vor der Berechnung des Ähnlichkeitswerts zweier Modelle werden die Vektorelemente mit ihren entsprechenden Singulärwerten skaliert, wie in (Deerwester u. a. 1990; Martin und Berry 2007) beschrieben. Der Ähnlichkeitsgrad zweier Modelle bzw. Vektoren basiert dabei auf der Kosinus-Ähnlichkeit (van Rijsbergen 1979, S. 25), sodass der Winkel zwischen zwei Modellvektoren zur Bestimmung des Ähnlichkeitsgrads herangezogen wird. Die Kosinus-Ähnlichkeit ist folgendermaßen definiert:

$$cos_{sim}(d_1, d_2) = \frac{d_1 \cdot d_2}{|d_1| \cdot |d_2|} = \frac{\sum_{i=1}^{n} d_{i1} \cdot d_{i2}}{\sqrt{\sum_{i=1}^{n} (d_{i1})^2} \cdot \sqrt{\sum_{i=1}^{n} (d_{i2})^2}}. \tag{5.3}$$

   Im Zähler wird das Skalarprodukt der Vektoren d_1 und d_2 berechnet, während im Nenner das Produkt der euklidischen Längen der Vektoren bestimmt wird. Daraus ergibt sich ein Wertebereich der Kosinus-Ähnlichkeit von $[-1, 1]$, wobei der Wert 1 für die höchstmögliche Ähnlichkeit der Vektoren steht und -1 für die größtmögliche Differenz. In einem Vektorraummodell entspricht der Wert 1 somit einem Winkel von $0°$, der Wert -1 einem Winkel von $180°$. In den LS3-Verfahren wird die Kosinus-Ähnlichkeit verwendet – und beispielsweise kein Distanzmaß wie die euklidische Distanz –, um einen Einfluss von Vektorlängen zu vermeiden (Turney und Pantel 2010).

   Um Ähnlichkeitswerte im üblicherweise verwendeten Intervall $[0, 1]$ zu erhalten, wird eine transformierte Kosinus-Ähnlichkeit berechnet, um den letztendlichen Ähnlichkeitsgrad zweier Modelle m_x und m_y zu bestimmen (Formel 5.4). Je kleiner der Ähnlichkeitswert ist, desto unähnlicher sind sich zwei Modelle; je größer der Wert ist, umso ähnlicher sind sie sich. Der Wert 1 bedeutet jedoch nicht, dass zwei Modelle identisch sind, da sich z. B. das Ablaufverhalten, die Graphstruktur oder teilweise die Beschriftungen unterscheiden können.

$$LSSM(m_x, m_y) = \frac{cos_{sim}(m_x, m_y) + 1}{2} \tag{5.4}$$

   Zur Berechnung der $n \times n$ großen Ähnlichkeitsmatrix $SimM$ wird Formel 5.4 schließlich für alle Kombinationen von Modellen bzw. Vektoren angewandt.

Die Matrixeinträge $SimM_{xy}, 1 \leq x, y \leq n$ entsprechen demnach den Ähnlichkeitswerten $LSSM(m_x, m_y)$. Zur Verdeutlichung siehe auch Schritt 4 in Abbildung 5.1. In Algorithmus 5.1 wird die Matrix $SimM$ in Zeile 5 berechnet.

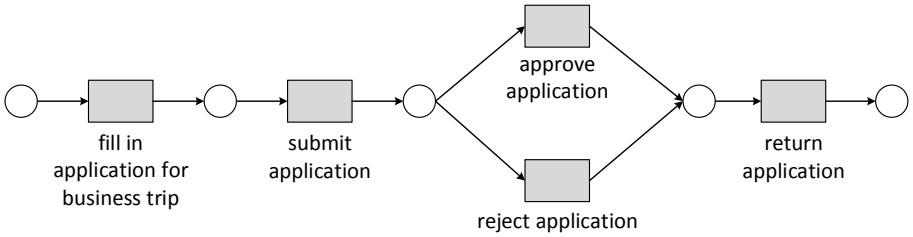
5. **Berechnung der Anfrageergebnisse:**
   Nach der Erstellung der Ähnlichkeitsmatrix $SimM$ erfolgt die Berechnung aller ähnlichen Modelle in einer Prozessmodellbibliothek durch einen Vergleich der Ähnlichkeitswerte mit einem Schwellenwert θ ($0 \leq \theta \leq 1$). Das Ergebnis einer ähnlichkeitsbasierten Suche zur Ermittlung aller ähnlichen Modelle $QR(M)$ enthält demzufolge eine Menge an Tupeln $(m_x, \{m_y\})$, die zu jedem Modell m_x die Menge an ähnlichen Modellen $\{m_y\}$ entsprechend Formel 5.5 angeben (die Berechnung erfolgt entsprechend der Zeilen 6–15 aus Algorithmus 5.1):

$$QR(\mathbf{M}) = \{(m_x, \{m_y\}) \mid SimM(m_x, m_y) \geq \theta\} \quad \forall\, x, y, 1 \leq x, j \leq n. \tag{5.5}$$

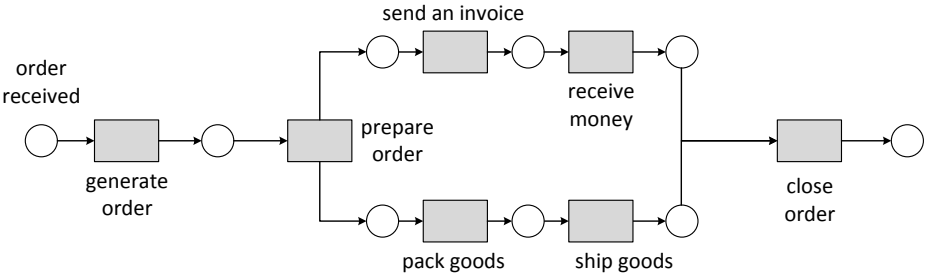
5.3.1 Beispiel für die QueryAll-Berechnung

Im Folgenden wird beispielhaft die Berechnungsweise des QueryAll-Verfahrens illustriert. Dazu werden die Modelle aus den Abbildungen 1.1 und 4.1 verwendet. Diese wurden auf Englisch übersetzt, da die prototypische Implementierung die Wortstammreduktion für Deutsch nicht umfasst. Die Modelle sind erneut in Abbildung 5.2 dargestellt. Darin sind zwei Modelle für einen Warenversandprozess sowie ein Modell eines Prozesses zur Beantragung einer Dienstreise abgebildet, sodass die beiden Warenversandmodelle ausreichend ähnlich sind, um jeweils im Ergebnis der beiden zugehörigen Anfragen enthalten zu sein. Das Dienstreisemodell sollte nicht als Ergebnis bei diesen beiden Anfragen enthalten sein, sondern nur bei der Verwendung als Anfrage selbst ausgegeben werden.

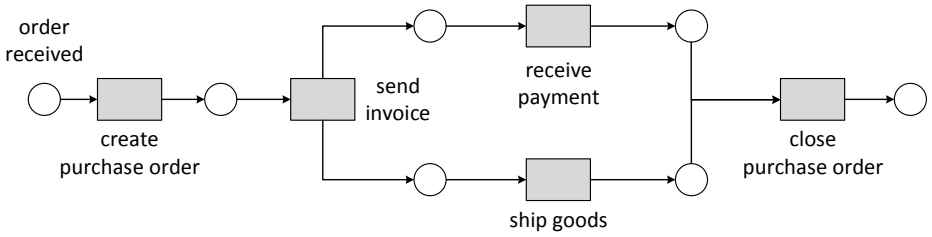
In Abbildung 5.3 sind die bei der Ergebnisberechnung erzeugten Term-Dokument und LSSM Matrizen abgebildet. Abbildung 5.3a zeigt die 21 Terme sowie ihre absoluten Häufigkeiten der drei Beispielmodelle, die durch die Vorverarbeitungsschritte Kleinschreibung, Entfernung von Stoppwörtern und Wortstammreduktion bestimmt wurden. Dabei wurde z. B. das Stoppwort „an“ aus Modell 2 entfernt. Zudem werden Terme mit dem gleichen Wortstamm durch die Wortstammreduktion darauf zurückgeführt. So wird aus den Termen „received“ und „receive“ in Modell 3 der Term „receiv“ durch den Porter-Stemming Algorithmus generiert.



(a) Beispielmodell 1: Modell für die Beantragung einer Dienstreise



(b) Beispielmodell 2: Modell für den Warenversand



(c) Beispielmodell 3: Modell für den Warenversand

Abb. 5.2: Modelle für die Beispielberechnung des LS3-QueryAll Algorithmus

Abbildung 5.3b zeigt schließlich die aus der Berechnung mit $k = 2$ resultierende LSSM Matrix der Ähnlichkeitswerte. Die Anzahl der Dimensionen k ist für diese kleine Beispielrechnung nur aus der Menge $\{1, 2, 3\}$ wählbar, da lediglich drei Modelle vorhanden sind. Da die Dimensionalität 1 nur die binäre Unterscheidung zwischen ähnlich und unähnlich zulässt, wurde $k = 2$ gewählt. Die Ähnlichkeitswerte unterscheiden sich deutlich zwischen den Modellen 2 und 3 sowie Modell 1, sodass beispielsweise mit einem Schwellenwert von $\theta = 0,8$ Modell 1 nur im Ergebnis enthalten ist, wenn

	M_1	M_2	M_3
$return$	1,0	0,0	0,0
$applic$	5,0	0,0	0,0
$reject$	1,0	0,0	0,0
$fill$	1,0	0,0	0,0
$vacat$	1,0	0,0	0,0
$submit$	1,0	0,0	0,0
$approv$	1,0	0,0	0,0
$order$	0,0	4,0	3,0
$receiv$	0,0	2,0	2,0
$pack$	0,0	1,0	0,0
$good$	0,0	2,0	1,0
$ship$	0,0	1,0	1,0
$close$	0,0	1,0	1,0
$monei$	0,0	1,0	0,0
$gener$	0,0	1,0	0,0
$prepar$	0,0	1,0	0,0
$send$	0,0	1,0	1,0
$invoic$	0,0	1,0	1,0
$purchas$	0,0	0,0	2,0
$creat$	0,0	0,0	1,0
$payment$	0,0	0,0	1,0

(a) Term-Dokument Matrix

	M_1	M_2	M_3
M_1	1,0	0,5	0,5
M_2	0,5	1,0	1,0
M_3	0,5	1,0	1,0

(b) LSSM Matrix

Abb. 5.3: Berechnungsmatrizen des LS3-QueryAll Algorithmus zu den Beispielmodellen

Modell 1 auch als Anfrage verwendet wird. Eine ausführliche Berechnung der Ergebnisse mit allen Zwischenschritten findet sich in Anhang A.2.1.

5.3.2 Berechnung von Ergebnissen für ein Anfragemodell

Die Berechnung von Anfrageergebnissen basierend auf einem Anfragemodell q ist in Algorithmus 5.2 abgebildet (LS3-Query Ansatz). Die ersten vier Zeilen stellen dabei die gleichen Schritte dar, die zuvor für den Fall zur Berechnung aller Anfrageergebnisse einer Modellbibliothek beschrieben wurden. Das heißt, dass eine gewichtete Term-Dokument Matrix erstellt wird, die dann mit Hilfe der Singulärwertzerlegung aufgeteilt und auf k Dimensionen projiziert wird. Die restlichen Zeilen stellen drei neue Schritte zur Berechnung eines Anfrageergebnisses dar.

1. **Erzeugung eines Pseudo-Dokuments:**
   Um ähnliche Modelle in einer Modellbibliothek zu finden, muss ein Anfragemodell q in den k-dimensionalen Vektorraum, der mit Hilfe der LSA berechnet wurde, projiziert werden. Daher wird ein sogenanntes *Pseudo-Dokument* erzeugt (Zeile 5 in Algorithmus 5.2), das einen Dokumentenvektor nach Definition 5.1

Algorithmus 5.2 : LS3-Query(**M**,q,t,k) nach (Schoknecht und Oberweis 2017)

input : Model library M, query model q, threshold t, dimensionality k.
output : A set of models QR containing the models similar to q.

```
   /* Calculate reduced singular value decomposition */
1  tdm = calculateTDMatrix(M);
2  wtdm = weightTDMatrix(tdm);
3  svd = calculateSVD(wtdm);
4  svdk = calculateReducedSVD (svd,k);
   /* Calculate pseudo document and similarity values */
5  pseudoDoc = generatePseudoDoc (q,svdk);
6  simValues[] = calculateLSSM(svdk,pseudoDoc);
   /* Calculate query results */
7  QR = ∅;
8  for (i = 0 to simValues.length − 1) do
9      if (SimValues[i] ≥ θ) then
10         Add model m_i to QR;
11     end
12 end
13 return QR;
```

darstellt. Nach (Deerwester u. a. 1990) kann ein Pseudo-Dokument eines Anfragemodells q entsprechend Formel 5.6 berechnet werden:

$$pseudoDoc = q^T U_k \Sigma_k^{-1}. \tag{5.6}$$

Dabei ist q^T ein mit der Log-Entropie gewichteter Vektor der Termhäufigkeiten in Anfragemodell q. Die Häufigkeiten von weiteren Termen in der Modellbibliothek **M**, die nicht in q vorkommen, erhalten den Wert 0. U_k und Σ_k sind die k-dimensionalen Term- und Singulärwert-Matrizen aus der LSA-Berechnung.

Zwei weitere Aspekte müssen in Bezug auf q^T beachtet werden. (1) Terme, die in q enthalten sind, aber nicht in **M** vorkommen, werden nicht berücksichtigt. Somit wird die ursprüngliche Term-Dokument Matrix von **M** nicht verändert. Dies ist sinnvoll, da q in den existierenden Vektorraum projiziert werden soll, um ähnliche Modelle zu finden und q hingegen nicht zur Modellbibliothek **M** hinzugefügt werden soll. (2) Bezüglich der Gewichtung von Termen in q durch das Log-Entropie Gewichtungsschema wurden die absoluten Termhäufigkeiten für die Berechnung nicht geändert. Die Terme werden demzufolge mit der gleichen Termhäufigkeit und Dokumentenanzahl gewichtet wie bei der ursprünglichen Berechnung der gewichteten Term-Dokument Matrix von **M**. Der Grund

dafür ist erneut, dass q in den existierenden Vektorraum projiziert werden soll, ohne **M** zu verändern.

2. **Berechnung von Ähnlichkeitswerten:**
   Nach der Erzeugung eines Pseudodokuments aus q werden die LSSM Ähnlichkeitswerte zwischen $pseudoDoc$ und allen Modellen $m \in \mathbf{M}$ berechnet (Zeile 6 in Algorithmus 5.2). Diese Berechnung wird wie zuvor in Formel 5.4 beschrieben durchgeführt.

3. **Finden ähnlicher Modelle:**
   Schließlich wird die Ergebnismenge der zu q ähnlichen Modelle durch die Zeilen 7–12 in Algorithmus 5.2 bestimmt. Jedes Modell in **M**, das einen gleich hohen oder größeren $LSSM$ Ähnlichkeitswert als θ zu q aufweist, wird zu der Ergebnismenge QR hinzugefügt:

$$QR(q, \mathbf{M}) = \{m_i \mid LSSM(q, m_i) \geq \theta\} \quad \forall\, m_i \in \mathbf{M}. \tag{5.7}$$

Die reduzierte Singulärwertzerlegung muss zudem nur ein Mal für eine Prozessmodellbibliothek berechnet werden, um eine beliebige Anzahl von Anfragen zu beantworten, solange die Bibliothek nicht verändert wird. Im Falle des Einfügens neuer Modelle, des Löschens oder der Änderung existierender Modelle muss hingegen der semantische Vektorraum neu berechnet werden, um die geänderten Modelldaten für eine Suche berücksichtigen zu können.

5.3.3 Einfügen und Löschen von Modellen

Wenn eine Prozessmodellbibliothek durch Einfügen, Löschen oder Überarbeitung eines Modells verändert wird, können die LS3-Suchansätze die Anpassungen der Bibliothek nicht direkt bei der Berechnung von Anfrageergebnissen berücksichtigen. Da der semantische Suchraum einmalig aus einer Term-Dokument Matrix erzeugt wird, wird ein Mechanismus benötigt, der die Änderungen an einer Bibliothek im Sinne enthaltener Terme und Termhäufigkeiten in den semantischen Vektorraum überträgt. Um dies zu erzielen, werden im Folgenden Algorithmen beschrieben, die auf aus einer Prozessmodellbibliothek erstellten Term-Dokument Matrix basieren und diese entsprechend anpassen. Daran anschließend muss die neu erstellte Term-Dokument Matrix erneut verarbeitet werden, um LSSM-Ähnlichkeitswerte zu berechnen

und Anfragen mit Hilfe des semantischen Vektorraums beantworten zu können. Das heißt, es müssen erneut die Zeilen 2–5 aus Algorithmus 5.1 ausgeführt werden.

Es wird somit ein Ansatz gewählt, der in Fällen von nebenläufiger Ausführung von Anfragen und Bibliotheksänderungen die Geschwindigkeit von Anfrageberechnungen und die Genauigkeit von Anfrageergebnissen zu balancieren versucht. Die Algorithmen zur Aktualisierung eines semantischen Vektorraums einer Prozessmodellbibliothek operieren dabei jeweils auf einer Term-Dokument Matrix, wodurch Geschwindigkeitsnachteile bei Änderungen einer Bibliothek begrenzt werden. Es müssen so nur neu eingefügte oder geänderte Modelle geparst und verarbeitet werden, um die darin enthaltenen Terme und Termhäufigkeiten zu ermitteln. Die Term-Dokument Matrix wird basierend auf diesen Daten adaptiert. Im Falle der Entfernung eines Modells wird sogar nur die Information benötigt, welches Modell aus der Bibliothek gelöscht wurde.

Der Pseudocode in Algorithmus 5.3 zeigt das Verfahren zur Anpassung einer Term-Dokument Matrix (TDM) für den Fall, dass ein neues Modell einer Bibliothek hinzugefügt wird. Als erstes werden die in dem neuen Modell m enthaltenen Terme und ihre jeweilige Anzahl extrahiert und in einer Hashmap gespeichert (Zeile 3). Anschließend werden in den Zeilen 6–15 die Termhäufigkeitseinträge der Vektordarstellung von m entsprechend Definition 5.1 berechnet. Für jeden in der ursprünglichen Term-Dokument Matrix enthaltenen Term, der ebenfalls in m enthalten ist, erhält das entsprechende Vektorelement die Termhäufigkeit in Modell m (Zeile 9) und der Term wird aus der in Zeile 3 erstellten Multimenge entfernt (Zeile 10). Anderenfalls wird das Vektorelement 0 gesetzt (Zeile 13), denn in diesem Fall enthält m den Term aus der TDM nicht. Daraufhin wird ein neuer Spaltenvektor für das Modell m der Term-Dokument Matrix hinzugefügt (Zeile 16).

Da ein neues Modell potentiell Terme enthalten kann, die noch nicht in der TDM vorhanden sind, müssen diese der TDM hinzugefügt werden (Zeilen 17–25). Für jeden Term, der noch in der Multimenge von m enthalten ist, wird ein neuer Zeilenvektor zur TDM hinzugefügt (Zeile 20). Die Vektor-Einträge sind 0, sollte die Spalte einem anderen Modell als m entsprechen. Wenn die Spalte hingegen dem Modell m entspricht, so wird die Termhäufigkeit aus m eingesetzt (Zeile 24). Schließlich wird die angepasste TDM in Zeile 26 zurückgegeben.

Algorithmus 5.4 zeigt das Vorgehen für die Entfernung eines existierenden Modells m aus einer Prozessmodellbibliothek. Wie bereits im Fall des Einfügens eines Modells,

Algorithmus 5.3 : InsertModel(TDM,m) nach (Schoknecht und Oberweis 2017)

input : Term-Document Matrix $TDM[i][j]$, model m.
output : Output Term-Document Matrix with m included.

```
1  int n = amountColumns (TDM);
2  int t = amountTerms (TDM);
3  HashMap<String,Integer> termsM = extractTerms(m);
4  String[] termsTDM = getTerms(TDM);
5  int[t] mVector;
   /* Calculate vector representation of m */
6  for (i = 0 to t − 1) do
7      term = termsTDM[i];
8      if (termsM.contains(term)) then
9          mVector[i] = termsM.get(term);
10         termsM.remove(term);
11     end
12     else
13         mVector[i] = 0
14     end
15 end
16 TDM = addColumn (TDM, mVector);
   /* Add additional terms of m */
17 int counter = 0;
18 for each (term in termsM) do
19     counter++;
20     TDM = addRow (TDM, term);
21     for (j = 0 to n − 1) do
22         TDM[t+counter][j] = 0;
23     end
24     TDM[t+counter][n] = termsM.count(termsM.get(term));
25 end
26 return TDM;
```

muss die TDM der Bibliothek angepasst werden. Dazu wird zunächst die Spalte der TDM entfernt, die dem Modell m entspricht (Zeile 1). Anschließend wird überprüft, ob ein Termvektor (Zeilenvektor) nicht mehr benötigt wird, da er lediglich Einträge mit dem Wert 0 enthält. Wenn alle Einträge gleich 0 sind, bedeutet das, dass der entsprechende Term in keinem der verbliebenen Modelle der Bibliothek mehr enthalten ist. Die Suche nach solchen Termvektoren sowie die zugehörige Entfernung des Vektors wird in den Zeilen 2–17 durchgeführt. Schließlich wird in Zeile 18 die angepasste TDM zurückgegeben.

Eine Änderung eines Modells wird als Entfernung des ursprünglichen Modells entsprechend Algorithmus 5.4 und des anschließenden Einfügens des geänderten Modells in die Modellbibliothek entsprechend Algorithmus 5.3 aufgefasst. Folglich ist

Algorithmus 5.4 : DeleteModel(TDM,m) nach (Schoknecht und Oberweis 2017)

input : Term-Document Matrix $TDM[i][j]$, model m.
output : Output Term-Document Matrix with m removed.

```
1  TDM = removeColumn (TDM, m);
2  int n = amountColumns (TDM);
3  int t = amountTerms (TDM);
4  boolean isNull = true;
5  for (i = 0 to t − 1) do
6      for (j = 0 to n − 1) do
7          if (TDM[i][j] ≠ 0) then
8              isNull = false;
9          end
10     end
11     if (isNull == true) then
12         TDM = removeRow (TDM, i);
13         t = t − 1;
14         i = i − 1;
15     end
16     isNull = true;
17 end
18 return TDM;
```

eine Änderung lediglich die Kombination der Algorithmen zur Entfernung und zum Einfügen von Modellen, die zuvor beschrieben wurden.

Prinzipiell könnte der semantische Vektorraum auch mit Hilfe von existierenden Vorschlägen aus der LSA Literatur angepasst werden. Diese *Folding-In* und *Folding-Out* Techniken (Berry u. a. 1995) sind möglicherweise sogar effizienter zu berechnen als die eben vorgestellten Varianten, da dabei der semantische Vektorraum nicht neu berechnet werden muss. Allerdings reduzieren diese Techniken die Genauigkeit der Ergebnisberechnung, weshalb sie nicht eingesetzt werden. Da die Evaluation in Kapitel 5.4 allerdings zeigt, dass zumindest für mittelgroße Modellbibliotheken die Berechnung des semantischen Vektorraums unter einer Sekunde beim Einsatz von handelsüblicher Hardware möglich ist und demzufolge der Einsatz der LS3-Verfahren in praktischen Anwendungsfällen nicht behindert wird, wird zugunsten der genaueren Ergebnisberechnung auf die Verwendung von Folding-Techniken verzichtet.

5.4 Evaluation

In diesem Teilkapitel wird zunächst in Abschnitt 5.4.1 auf die verwendeten Modelldatensätze und deren Charakteristika eingegangen. Des Weiteren werden die verglichenen Suchverfahren und der Ablauf der Evaluation erläutert. Daraufhin werden in Abschnitt 5.4.2 die Ergebnisse der Evaluation beschrieben, die abschließend in Kapitel 5.4.3 hinsichtlich mehrerer Aspekte diskutiert werden.

5.4.1 Testumgebung

Für die empirische Evaluation der LS3-Suchverfahren werden drei Modelldatensätze verwendet. Der erste wurde von Vogelaar u. a. (2011) eingeführt. Dieser Datensatz enthält Modelle von acht verschiedenen Geschäftsprozessen aus zehn unterschiedlichen niederländischen Gemeinden (Dutch Municipalities, DM). Somit besteht dieser Datensatz aus insgesamt 80 Modellen. Die Modelle sind linguistisch harmonisiert, d. h., dass die Beschriftungen von Modellelementen eindeutig und konsistent sind. Daher sind semantisch äquivalente Aktivitäten auf die gleiche Weise beschriftet, was ähnlichkeitsbasierten Suchverfahren entgegenkommen sollte, die auf einem Prozessmodell-Matching basieren.

Der zweite Modelldatensatz enthält die bereits zur Evaluation der Triple-S Ansätze in Kapitel 4 verwendeten Modelle aus dem Process Model Matching Contest 2015 (Antunes u. a. 2015). Auch für die Evaluation der LS3-Verfahren werden die Universitätszulassungsmodelle (University Admission, UA) und die Geburtsprozessmodelle (Birth Registration, BR) verwendet. Diese umfassen wie in Kapitel 4.4 beschrieben jeweils neun Modelle, wobei deren Beschriftungen nicht linguistisch harmonisiert sind. Das bedeutet, dass die Modelle zwar semantisch gleiche Aktivitäten enthalten, diese jedoch mit Beschriftungen versehen sind, die eine andere Wortwahl aufweisen. Insgesamt werden daher 18 Modelle für die Evaluation verwendet. Der dritte Modelldatensatz enthält schließlich darüber hinaus noch jeweils zehn Modelle von vier Prozessen. Diese stammen aus Fortbildungsveranstaltungen von Camunda[3] (Camunda Models, CM), bei denen die Teilnehmer zu textuellen Beschreibungen ein zugehöriges Prozessmodell erstellen mussten. Aus den zahlreichen Lösungsmodellen wurden für diese Evaluation für die vier Prozesse jeweils zehn zufällig ausgewählt. Auch bei diesen

[3] Die Modelle sind unter https://github.com/camunda/bpmn-for-research verfügbar.

Modellen sind heterogene Beschriftungen enthalten, da die Modelle von verschiedenen Teilnehmern der Fortbildung erstellt wurden. Daher gestaltet sich das Auffinden von korrekten Matches wie bei den Modellen des Process Model Matching Contest 2015 ebenso als schwierig.

Für die verschiedenen verwendeten Modellsammlungen sind weitere Charakteristika in Tabelle 5.3 angegeben. Die jeweilige Anzahl an Termen wurde dabei nach der Stoppwort Entfernung und Wortstammreduktion berechnet. Entsprechend der Beschreibung von Thaler u. a. (2017) werden für den als *Mined Models* klassifizierten DM-Datensatz gute Ergebnisse für Matching-basierte Verfahren erwartet, wohingegen für die UA- und BR-Datensätze (*Field Models*) von weniger guten Ergebnissen ausgegangen wird. Für die CM-Modelle, die zur Kategorie *Controlled Models* gehören, sollten die Ergebnisse zwischen den beiden anderen Kategorien liegen.

Tab. 5.3: Charakteristika der LS3 Evaluationsdatensätze

	DM	UA	BR	CM	Gesamt
Anzahl Prozesse	8	1	1	4	14
Anzahl Modelle pro Prozess	10	9	9	10	-
Gesamtanzahl Modelle	80	9	9	40	138
Gesamtanzahl unterschiedlicher Terme[a]	391	149	141	191	656
∅ Anzahl Terme pro Modell	78,08	71,22	61,56	36,25	64,43
STD Anzahl Terme	49,08	32,46	17,78	10,94	42,98
Min. Anzahl Terme	19	38	37	21	19
Max. Anzahl Terme	236	134	87	68	236

[a] Die Anzahl unterschiedlicher Terme bezieht sich auf den englischen Begriff „distinct“, sodass ein in mehreren Modellen vorkommender Term nur einfach gezählt wird. Jeweils in Bezug auf alle Modelle eines Datensatzes.

Zur Evaluation wurden alle Modelle aus der ursprünglichen Repräsentation manuell in Petri-Netze transformiert und als PNML-Dateien (siehe Kapitel 2.1.2) gespeichert. Ein Prototyp,[4] der die LS3-Ansätze realisiert, wurde in Java entwickelt, wobei zur Tokenisierung der Beschriftungen der Stanford Parser (Klein und Manning 2003) eingesetzt wurde. Die Evaluation wurde mit einem Laptop mit der folgenden Spezifikation durchgeführt: Intel(R) i7-4750HQ CPU, 8 GB RAM, Windows 8.1 und JAVA 1.8.

[4] Der Quellcode steht unter https://github.com/ASchoknecht/LS3 zum Download bereit.

In dem dreistufigen Evaluationsverfahren (siehe auch Abbildung 5.4) wurde zunächst lediglich der DM-Datensatz genutzt und Precision, Recall und F-Wert bestimmt, indem jedes Modell des DM-Datensatzes als Anfragemodell verwendet wurde. Anschließend wurden die zurückgelieferten Modelle gegenüber einem Goldstandard bewertet, in dem für jedes Anfragemodell die relevanten Ergebnismodelle festgehalten waren. Im Kontext dieser Evaluation ist ein zurückgegebenes Modell relevant, wenn es den gleichen Prozess aus dem DM-Datensatz repräsentiert wie das Anfragemodell.[5] Zu jedem Anfragemodell wurden somit im besten Fall genau die zehn Modelle zurückgeliefert, die den gleichen zugrundeliegenden Prozess darstellen. Die als Anfrage verwendeten Modelle wurden also nicht aus dem Datensatz entfernt.

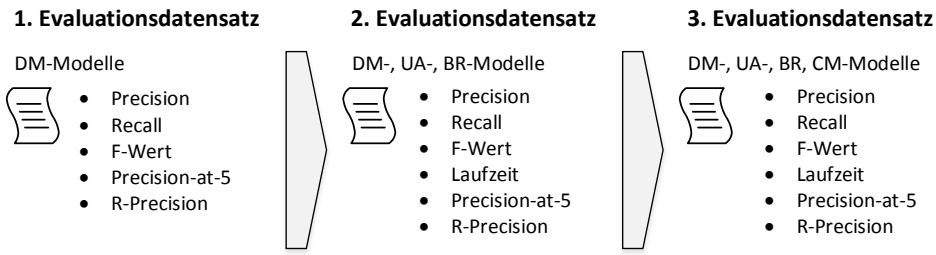


Abb. 5.4: Überblick über den Verlauf der dreistufigen Evaluation. Beschrieben sind die in den Stufen verwendeten Modelle sowie die berechneten Kennzahlen.

Für Precision, Recall und F-Wert wurden jeweils Macro- und Micro-Average Werte sowie die Standardabweichung berechnet. Die mit diesem Datensatz ermittelten Werte stellen gewissermaßen die Basis für die folgenden Stufen dar, da die Ähnlichkeitsberechnung aufgrund der harmonisierten Beschriftungen im Vergleich zu den anderen beiden Datensätzen einfach ist. Insbesondere gestaltet sich die Berechnung von Matches einfach, da semantisch gleiche Aktivitäten auch gleich benannt sind (Thaler u. a. 2017). Dadurch ist zu erwarten, dass vor allem Verfahren, die einen Ähnlichkeitswert basierend auf Matches bestimmen, gute Ergebnisse erzielen können.

In der zweiten Stufe wurden zu dem DM-Datensatz alle Modelle aus dem Matching Wettbewerb hinzugefügt, sodass jetzt 98 Modelle für die Evaluation verwendet wur-

[5] Der Goldstandard wurde daher nicht von Experten bestimmt, sondern aufgrund des Aufbaus des Datensatzes. Dies gilt auch für die weiteren Evaluationsstufen.

den. In der dritten Stufe wurden schließlich noch die Modelle aus dem dritten Datensatz von Camunda hinzugefügt, sodass 138 Modelle für die Evaluation eingesetzt wurden. Durch das schrittweise Hinzufügen von immer mehr Modellen, für die Matches schwierig zu berechnen sind, soll eine Übersicht über die Entwicklung der Qualität der Suchverfahren gegeben werden. Für die 2. und 3. Stufe der Evaluation wird erwartet, dass Matching-basierte Verfahren schlechter abschneiden als die LS3-Verfahren. Neben den zuvor berechneten Kennzahlen werden zudem die Laufzeiten zur Berechnung der Anfrageergebnisse mit angegeben.

Für die Laufzeitberechnung im LS3-QueryAll Fall beinhaltet dies die Erstellung der Term-Dokument Matrix, die anschließende Singulärwertzerlegung und Ähnlichkeitsberechnung sowie die letztendliche Ergebnisberechnung für alle Modelle. Im LS3-Query Fall berücksichtigt die Zeitmessung die Erstellung eines Pseudodokuments und die letztendliche Ergebnisberechnung. Hierbei wird demnach die Zeit zur Erstellung des semantischen Vektorraums der Modellbibliothek nicht berücksichtigt, sondern davon ausgegangen, dass ein solcher bereits berechnet wurde. Somit liegt der Fokus auf der Geschwindigkeit zur Ergebnisberechnung einer Anfrage.

Da diese ähnlichkeitsbasierte Suchfunktion für die Wiederverwendung von Prozessmodellen gedacht ist, wird darüber hinaus die Reihenfolge der Suchergebnisse durch die Berechnung von Precision-at-n und R-Precision ergänzt. Vorteilhaft an dieser Art der Qualitätsbestimmung ist, dass für die Bewertung keine Schwellenwerte bestimmt werden müssen, sondern die Anfrageergebnisse absteigend nach dem Grad der Ähnlichkeit sortiert sind. Demzufolge entsprechen z. B. für ein Anfragemodell die ersten zehn Ergebnismodelle den Modellen mit den zehn höchsten Ähnlichkeitswerten.

Zusätzlich werden die für die LS3-Ansätze berechneten Werte mit fünf anderen Ansätzen für eine ähnlichkeitsbasierte Suche nach Prozessmodellen verglichen. Zu den verglichenen Ansätzen gehören *CF* (van Dongen u. a. 2008), *GEDS* (Dijkman u. a. 2009a), *SSBOCAN* (Akkiraju und Ivan 2010), *LAROSA* (La Rosa u. a. 2010), und *FBSE* (Yan u. a. 2012c). Van Dongen u. a. (2008) verwenden dabei *Causal Footprints*, die Reihenfolgenbeziehungen zwischen Aktivitäten in Prozessmodellen beschreiben. Die zugrundeliegenden Matches wurden in der folgenden Evaluation durch gleich benannte Aktivitäten bestimmt, während die weiteren in der Publikation beschriebenen Parameter wie darin angegeben festgelegt wurden.

Dijkman u. a. (2009a) nutzen ein Ähnlichkeitsmaß basierend auf einer Graph-Editier Distanz, um Modellsammlungen zu durchsuchen. In der im Folgenden beschriebe-

nen Evaluation wurde das Konzept der Editier-Distanz sowohl auf Beschriftungen (String-Editier Distanz) als auch auf die Graphstruktur von Modellen (Graph-Editier Distanz) angewendet. Zur Optimierung der Ähnlichkeitsmatrix wurde der beschriebene Greedy-Algorithmus verwendet und die drei Parameter aus (Dijkman u. a. 2009a) wurden gleich gewichtet

In (Akkiraju und Ivan 2010) wird ein Ähnlichkeitswert zwischen Prozessmodellen auf Basis der Anzahl identisch beschrifteter Aktivitäten bestimmt. La Rosa u. a. (2010) erweitern den Ansatz aus (Dijkman u. a. 2009a), indem auch Kontrollflusskonnektoren berücksichtigt werden und indem das Matching nicht nur auf der String-Editier-Distanz, sondern zusätzlich auf einem linguistischen Ähnlichkeitsmaß beruht. Es wurde die ursprüngliche Implementierung mit Parameterwerten aus der Veröffentlichung verwendet. Yan u. a. (2012c) berechnen die Ähnlichkeit von Beschriftungen ebenfalls mit Hilfe einer String-Editier Distanz und berücksichtigen zudem Graph-strukturelle Aspekte, indem fünf Rollen zur Charakterisierung eines Knotens definiert werden. Die Schwellenwerte wurden wie in der ursprünglichen Publikation beschrieben festgelegt. Die resultierende Ähnlichkeitsmatrix wurde mit dem Greedy Algorithmus optimiert. Somit verwenden alle verglichenen Ansätze ein Matching von Prozessmodellelementen, um Ähnlichkeitswerte und Anfrageergebnisse zu berechnen.

Die Berechnungen für diese Ansätze wurden mit dem RefMod-Miner[6] Dienst durchgeführt, der Implementierungen der verglichenen Ansätze bietet. So konnten die Ähnlichkeitswerte aller Paare von Prozessmodellen berechnet werden, wobei anschließend die letztendlichen Anfrageergebnisse durch einen festgelegten Schwellenwert ermittelt wurden. Der Schwellenwert wurde zur Bestimmung von Precision, Recall und F-Wert für jeden Ansatz so festgelegt, dass der durchschnittliche F-Wert maximiert wird. Dazu wurden für jeden Ansatz die F-Werte für alle Schwellenwerte im Intervall $[0,0,1,0]$ berechnet (jeweils in Schritten von 0,01 Punkten) und der Schwellenwert mit dem höchsten durchschnittlichen F-Wert ausgewählt. Zur Berechnung von R-Precision und Precision-at-5 wurde der Schwellenwert auf 0,0 gesetzt, sodass sich eine nach absteigendem Ähnlichkeitswert sortierte Liste von Anfrageergebnissen ergab.

Für die LS3-Algorithmen wurde ebenso wie zuvor beschrieben verfahren. Zusätzlich wurden für die LS3-Verfahren noch die beste Dimensionalitätsanzahl bestimmt, in-

[6] http://rmm.dfki.de

dem alle Werte kleiner als der Rang r der Singulärwertzerlegung getestet wurden und die Dimensionalitätsanzahl mit dem besten durchschnittlichen F-Wert ausgewählt wurde. Dies bedeutet letztlich, dass für die LS3-Verfahren für alle Dimensionalitäten aus der Menge $\{1, ..., r\}$ und alle Schwellenwerte im Intervall $[0.0, 1.0]$ die Anfrageergebnisse berechnet wurden und die Kombination mit dem besten durchschnittlichen F-Wert für den Vergleich mit den anderen Ansätzen verwendet wurde.

5.4.2 Ergebnisse

Die Ergebnisse bezüglich der ersten Stufe der Evaluation sind in Tabelle 5.4 dargestellt. Die Tabelle zeigt die durchschnittlichen Werte für Precision, Recall und F-Wert aller 80 Anfragen sowie die zugehörigen Standardabweichungen. Die vorletzte Spalte gibt die Anzahl von Anfragen an, deren F-Wert gleich 1 ist und die damit ein optimales Ergebnis darstellen. Zur Berechnung der LS3-Ergebnisse wurde der Dimensionalitätsparameter auf $k = 14$ und der Schwellenwert θ auf 0,79 gesetzt.

Tab. 5.4: Performanz der verglichenen Suchansätze für den DM-Datensatz

	Precision			**Recall**			**F-Wert**				θ
	Mi[c]	Ma	STD	Mi	Ma	STD	Mi	Ma	STD	F = 1	
LS3-QueryAll	0,95	0,96	0,10	0,92	0,92	0,17	0,93	0,93	0,14	43	0,79
LS3-Query	**0,98**[b]	**0,99**	0,06	0,82	0,82	0,29	0,89	0,85	0,25	36	0,79
CF[a]	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.
GEDS	0,94	0,95	0,10	0,96	0,96	0,10	**0,95**	**0,95**	0,08	44	0,48
SSBOCAN	0,93	0,94	0,11	**0,98**	**0,98**	0,09	**0,95**	**0,95**	0,09	**51**	0,58
LAROSA	0,80	0,84	0,19	0,77	0,79	0,21	0,78	0,78	0,16	4	0,19
FBSE	0,38	0,42	0,18	0,72	0,72	0,24	0,50	0,50	0,18	0	0,88

[a] Für (van Dongen u. a. 2008) konnten keine Werte ermittelt werden, da ein Speicherüberlauf auftrat.
[b] Fett gedruckte Einträge kennzeichnen die besten Werte der entsprechenden Spalte.
[c] Mi = Micro-Average, Ma = Macro-Average, STD = Standardabweichung.

Die Ergebnisse zeigen, dass die LS3-Ansätze zwar sehr gute Resultate liefern, jedoch insbesondere der SSBOCAN Ansatz deren Qualität übertrifft. Die LS3-Ansätze sind mit Werten von 0,99 und 0,96 hinsichtlich der Precision-Werte die besten Verfahren, wobei GEDS (0,95) und SSBOCAN (0,94) mit ebenfalls sehr hohen Werten auf den Plätzen 3 und 4 liegen. Die beiden weiteren Ansätze, für die Ergebnisse ermittelt werden konnten, folgen mit deutlich geringeren Werten. Bezüglich der Recall- und F-Werte ergibt sich eine andere Reihenfolge. Jetzt liegen die SSBOCAN und GEDS Verfah-

ren vor den LS3-Ansätzen. Während zwar der LS3-QueryAll Ansatz nur geringfügig schlechtere Ergebnisse erzielt, erreicht der LS3-Query Ansatz vor allem in Bezug auf den Recall deutlich niedrigere Werte. Gegenüber dem besten Vergleichsansatz SSBOCAN (0,98) wird nur ein geringer Wert von 0,82 erzielt. Die F-Werte von SSBOCAN, GEDS und LS3-QueryAll liegen eng beieinander mit Werten von 0,95 und 0,95 bzw. 0,93. Auch hier liegt der LS3-Query Ansatz auf dem vierten Rang mit einem Wert von 0,85. Der SSBOCAN Ansatz berechnet auch die größte Anzahl an Anfragen mit einem F-Wert von 1. 51 der insgesamt 80 Anfragen enthalten genau die zehn korrekten Modelle im Ergebnis, wohingegen bei GEDS noch 44 und bei LS3-QueryAll 43 Anfragen einen F-Wert von 1 erzielen. Der LS3-Query Ansatz folgt wiederum auf Rang 4 mit 36 optimal beantworteten Anfragen. Zudem bedeuten die geringen Werte der Standardabweichungen, dass die einzelnen Ergebnisse eine geringe Streuung um den Mittelwert herum aufweisen. Insbesondere bei GEDS und SSBOCAN sind diese mit Werten unter 0,12 sehr gering, sodass sich eine durchgehend hohe Qualität der Anfrageergebnisse erkennen lässt. Für den CF-Ansatz ließen sich hingegen generell keine Werte berechnen, da es zu einem Speicherüberlauf bei der Berechnung der Ähnlichkeitswerte kam. Dies gilt entsprechend auch für die weiteren Evaluationen, da die für die Modelle jeweils erzeugte Datenmenge zu groß für den Hauptspeicher war.

Bei der zweiten Stufe der Evaluation wurden zu dem DM-Datensatz die UA- und BR-Modelle hinzugefügt, sodass sich insgesamt 98 Modelle ergaben. Hinsichtlich des Ablaufs gab es keine Änderungen zur vorherigen Evaluation. Es befanden sich erneut alle Modelle in der Prozessmodellbibliothek und jedes der 98 Modelle wurde als Anfrage verwendet. Wenn ein Anfragemodell aus dem DM-Datensatz stammte, sollten im besten Fall die zehn entsprechenden Modelle zurückgegeben werden; im Falle eines Modells aus dem UA- bzw. BR-Datensatz die entsprechenden neun Modelle. Zur Bestimmung der Ergebnisqualität wurden ebenfalls erneut die durchschnittlichen Werte von Precision, Recall und F-Wert sowie die Standardabweichung berechnet. Zusätzlich wurde die Laufzeit der verglichenen Ansätze bestimmt. Eine Übersicht der Ergebnisse für die Macro-Average Werte ist in Tabelle 5.5 angegeben. Tabelle 5.6 stellt die Micro-Average Werte dar. Für die Laufzeit ist dabei jeweils der durchschnittliche Wert von zehn Berechnungsdurchgängen angegeben.

Bei der zweiten Stufe der Evaluation ergaben sich Änderungen der Schwellenwerte für die LS3-Ansätze. Für den LS3-Query Ansatz lieferte sowohl zur Berechnung der Macro- als auch der Micro-Average Werte $\theta = 0,79$ die besten Ergebnisse, während für das LS3-QueryAll Verfahren für die Macro-Average Berechnung $\theta = 0,76$ und

Tab. 5.5: Performanz der verglichenen Suchansätze für die DM-, UA- und BR-Datensätze

Macro-Average	**Precision**		**Recall**		**F-Wert**			**Laufzeit**	θ
	AVG	STD	AVG	STD	AVG	STD	F = 1	AVG	
LS3-QueryAll	0,93	0,15	**0,87**	0,23	**0,87**	0,18	45	**1082,9 ms.**	0,76
LS3-Query	**0,98**[b]	0,07	0,78	0,31	0,82	0,25	47	**5,1 ms.**	0,79
CF[a]	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.
GEDS	0,96	0,09	0,83	0,30	0,84	0,25	44	106:59 Min.	0,48
SSBOCAN	0,96	0,09	0,82	0,32	0,83	0,28	**51**	85:07 Min.	0,58
LAROSA	0,84	0,20	0,69	0,29	0,70	0,23	4	67:39 Min.	0,18
FBSE	0,31	0,17	0,66	0,26	0,40	0,18	0	273:40 Min.	0,88

[a] Für (van Dongen u. a. 2008) konnten keine Werte ermittelt werden, da ein Speicherüberlauf auftrat.
[b] Fett gedruckte Einträge kennzeichnen die besten Werte der entsprechenden Spalte.

Tab. 5.6: Performanz der verglichenen Suchansätze für die DM-, UA- und BR-Datensätze

Micro-Average	**Precision**	**Recall**	**F-Wert**	**F = 1**	θ
LS3-QueryAll	**0,96**[b]	**0,85**	**0,90**	**51**	0,80
LS3-Query	**0,96**	0,79	0,86	47	0,79
CF[a]	n.a.	n.a.	n.a.	n.a.	n.a.
GEDS	0,94	0,84	0,89	44	0,48
SSBOCAN	0,93	**0,85**	0,89	**51**	0,58
LAROSA	0,80	0,68	0,73	4	0,19
FBSE	0,28	0,66	0,39	0	0,88

[a] Für (van Dongen u. a. 2008) konnten keine Werte ermittelt werden, da ein Speicherüberlauf auftrat.
[b] Fett gedruckte Einträge kennzeichnen die besten Werte der entsprechenden Spalte.

für die Micro-Average Berechnung $\theta = 0,8$ am besten abschnitten. Bezüglich des Dimensionalitätsparameters wurden für den LS3-QueryAll Ansatz die Berechnungen jeweils mit $k = 19$ durchgeführt, für den LS3-Query Ansatz mit $k = 14$. Die weiteren Schwellenwerte sind in den Tabellen angegeben.

Im Hinblick auf die Performanz der verschiedenen verglichenen Suchverfahren für die erweiterte Modellbibliothek fallen die Macro-Average Recall- und F-Werte für alle Verfahren geringer aus. Die Precision-Werte bleiben hingegen auf dem gleichen, sehr hohen Niveau. Die Werte der LS3-Ansätze sinken nur um drei bzw. einen Punkt auf 0,93 bzw. 0,98, während die Werte von GEDS und SSBOCAN sogar auf jeweils 0,96 steigen und LAROSA weiterhin bei 0,84 liegt. Lediglich der FBSE Ansatz verschlechtert sich deutlich von 0,42 auf 0,31. Die Recall-Werte sinken jedoch mehr oder weniger deutlich bei allen Verfahren. So weist der LS3-QueryAll Ansatz jetzt den höchsten Wert mit 0,87 auf, wohingegen die beiden zuvor besten Verfahren, GEDS und SSBOCAN, mehr als 0,1 Punkte verlieren. GEDS fällt von 0,96 auf 0,84 und SSBOCAN sogar von 0,98 auf 0,83. Der Wert des LS3-Query Ansatzes fällt nicht ganz so stark, verringert sich aber dennoch um 0,04 Punkte und erreicht noch einen Wert von 0,78. Die beiden letzten Verfahren, LAROSA und FBSE, liegen deutlich dahinter mit Werten von 0,69 und 0,66.

Hinsichtlich der F-Werte ergibt sich ein vergleichbares Bild wie für die Recall-Werte. Auch hier schneiden alle Verfahren aufgrund der geringeren Recall-Werte schlechter als zuvor ab. Dabei erzielt der LS3-QueryAll Ansatz ebenfalls den höchsten Wert mit 0,87, gefolgt von GEDS mit 0,84, SSBOCAN mit 0,83 und LS3-Query mit 0,82. Die beiden weiteren Verfahren liegen wiederum deutlich zurück mit Werten von 0,7 für LAROSA und 0,4 für FBSE. Somit erzielen die vier besten Ansätze beinahe gleich gute Ergebnisse, wobei die LS3-Verfahren besser mit den heterogenen Beschriftungen umgehen als GEDS und SSBOCAN, was sich in dem geringeren Rückgang der F-Werte widerspiegelt.

Auch bezüglich der Anzahl optimal beantworteter Anfragen lassen sich kaum Unterschiede feststellen. In dieser Kategorie schneidet SSBOCAN mit 51 optimal beantworteten Anfragen am besten ab. Der LS3-Query Ansatz liegt auf Platz zwei mit 47, gefolgt von LS3-QueryAll mit 45 und GEDS mit 44. Hierbei ist die Anzahl für die LS3-Ansätze sogar gestiegen im Vergleich zu den vorherigen Evaluationsergebnissen. Der Wert erhöht sich für den LS3-QueryAll Ansatz um 2 und für den LS3-Query Ansatz um 11. Bei GEDS und SSBOCAN bleibt der Wert hingegen gleich. Zudem steigen

die Standardabweichungen insbesondere für die Recall- und F-Werte, sodass ersichtlich wird, dass die Ergebnisqualität nicht mehr so konstant hoch ist wie in der ersten Stufe der Evaluation. Die größere Streuung der Werte lässt sich auf die UA- und BR-Modelle zurückführen, da für Anfragen mit diesen Modellen wesentlich schlechtere Ergebnisse bei allen Verfahren auftraten.

Der größte Unterschied zwischen allen Ansätzen lässt sich jedoch bei der benötigten Laufzeit zur Berechnung der Ergebnisse feststellen. Während die LS3-Verfahren lediglich etwas mehr als eine Sekunde zur Berechnung aller Anfrageergebnisse benötigen, braucht LAROSA als zweitschnellstes Verfahren schon ca. 67 Minuten. Das Ergebnis einer Anfrage ließ sich durch LS3-Query schon in durchschnittlich 5,1 Millisekunden berechnen.

In Bezug auf die Micro-Average Ergebnisse in Tabelle 5.6 zeigt sich ein leicht anderes Bild. Wenn die Berechnung der Anfrageergebnisse nicht mehr hinsichtlich des aggregierten Macro-Average F-Werts, sondern des detaillierteren Micro-Average F-Werts optimiert wird, schneidet der LS3-QueryAll Ansatz am besten ab. Er erzielt für alle Maßzahlen die höchsten Werte. GEDS und SSBOCAN folgen nun auf dem 2. Platz mit jeweils 0,89 als F-Wert und das LS3-Query Verfahren auf Platz 3 mit einem Wert von 0,86. Die gleiche Reihenfolge lässt sich auch für die Precision- und Recall-Werte erkennen. Des Weiteren kann für alle Verfahren eine Steigerung der Werte festgestellt werden, was auf die unterschiedliche Gewichtung der schlechteren Ergebnisse für die UA- und BR-Modelle bei der Micro- gegenüber der Macro-Average Berechnung zurückzuführen ist. Da bei der Berechnung der Micro-Average Werte kein Durchschnitt der einzelnen Anfrageergebnisse gebildet wird, sondern die einzelnen korrekten und falschen Ergebnisse aufsummiert werden, lässt sich zudem keine Standardabweichung berechnen (siehe auch Kapitel 3.4). Und auch die Laufzeit zur Ergebnisberechnung hat sich nicht geändert, weshalb diese nicht mehr angegeben wird.

Für die dritte Stufe der Evaluation wurden schließlich noch die Camunda Modelle dem Evaluationsdatensatz der zweiten Stufe hinzugefügt, sodass letztlich 138 Modelle in der Modellbibliothek vorhanden sind. Die Macro-Average Ergebnisse sind in Tabelle 5.7 abgebildet und die Micro-Average Ergebnisse finden sich in Tabelle 5.8. Der Dimensionalitätsparameter $k = 25$ lieferte für die LS3-Ansätze für diese Bibliothek die besten Ergebnisse. Die Schwellenwerte können wiederum aus den Tabellen abgelesen werden.

Tab. 5.7: Performanz der verglichenen Suchansätze für die DM-, UA-, BR- und CM-Datensätze

Macro-Average	Precision		Recall		F-Wert			Laufzeit	θ
	AVG	STD	AVG	STD	AVG	STD	F = 1	AVG	
LS3-QueryAll	0,98	0,07	**0,88**	0,24	**0,90**	0,19	**77**	**1551 ms.**	0,79
LS3-Query	**0,99**[b]	0,05	0,81	0,30	0,85	0,25	69	**4,5 ms.**	0,79
CF[a]	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.
GEDS	0,95	0,12	0,64	0,41	0,63	0,39	45	298:33 Min.	0,46
SSBOCAN	0,97	0,09	0,67	0,39	0,70	0,34	51	295:37 Min.	0,58
LAROSA	0,89	0,18	0,68	0,39	0,70	0,34	6	104:50 Min.	0,18
FBSE	0,26	0,14	0,52	0,23	0,33	0,15	0	681:52 Min.	0,92

[a] Für (van Dongen u. a. 2008) konnten keine Werte ermittelt werden, da ein Speicherüberlauf auftrat.
[b] Fett gedruckte Einträge kennzeichnen die besten Werte der entsprechenden Spalte.

Hinsichtlich der F-Werte bleiben die LS3-Ansätze auf dem gleichen, hohen Niveau wie bereits zuvor. So erzielt der LS3-QueryAll Ansatz einen Wert von 0,9, wohingegen das LS3-Query Verfahren immerhin noch einen Wert von 0,85 aufweist. Alle weiteren Verfahren erreichen jetzt deutlich geringere Werte. SSBOCAN und LAROSA als nächstbeste erreichen beide einen F-Wert von 0,7 und weisen damit schon eine Differenz von 0,15 bzw. 0,2 Punkten auf. Für die Recall-Werte ergibt sich die gleiche Reihenfolge und auch bezüglich der Precision-Werte sind die LS3-Verfahren auf den ersten beiden Plätzen. Allerdings ist der Abstand zum nächst besten Verfahren mit 0,01 bzw. 0,02 Punkten wesentlich geringer, da SSBOCAN mit 0,97 ebenfalls einen sehr hohen Wert aufweist. Auch die weiteren Ansätze erzielen mit Ausnahme von FBSE hohe Precision-Werte.

Die Werte der Standardabweichungen bleiben für die LS3-Ansätze und LAROSA auf gleichem Niveau, während sich die Werte der anderen Verfahren weiter erhöhen. Dies bedeutet, dass für diese Verfahren die Werte eine zunehmend größere Streuung um den Mittelwert aufweisen, sodass die Qualität der Anfrageergebnisse zunehmend variiert. Schließlich bleibt auch die Laufzeit für die LS3-Verfahren konstant. Es zeigt sich lediglich für LS3-QueryAll eine erhöhte Laufzeit aufgrund der größeren Menge an Modellen in dem Evaluationsdatensatz. Im Gegensatz dazu benötigten die anderen Verfahren deutlich über eine Stunde, um alle Ergebnisse zu berechnen. Schließlich gelten auch für die Micro-Average Ergebnisse in Tabelle 5.8 die gleichen Erkenntnisse, die für die Macro-Average Ergebnisse zuvor beschrieben wurden.

Tab. 5.8: Performanz der verglichenen Suchansätze für die DM-, UA-, BR- und CM-Datensätze

Mirco-Average	**Precision**	**Recall**	**F-Wert**	**F = 1**	θ
LS3-QueryAll	0,97	**0,88**	**0,93**	**77**	0,79
LS3-Query	**0,98**[b]	0,81	0,89	69	0,79
CF[a]	n.a.	n.a.	n.a.	n.a.	n.a.
GEDS	0,94	0,63	0,76	44	0,48
SSBOCAN	0,94	0,67	0,78	51	0,58
LAROSA	0,82	0,69	0,75	6	0,18
FBSE	0,23	0,53	0,32	0	0,92

[a] Für (van Dongen u. a. 2008) konnten keine Werte ermittelt werden, da ein Speicherüberlauf auftrat.
[b] Fett gedruckte Einträge kennzeichnen die besten Werte der entsprechenden Spalte.

In Tabelle 5.9 sind die durchschnittlichen Vergleichswerte für R-Precision und Precision-at-n dargestellt. Die Precision-at-n Werte sind jeweils für $k = 5$ berechnet, während für die Berechnung der R-Precision Werte für die DM- und CM-Modelle entsprechend $k = 10$ und für die UA- und BR-Modelle $k = 9$ gilt. Durch die Analyse dieser Kennzahlen kann eine Aussage getroffen werden, wie gut die Ergebnisqualität zu einer Anfrage ist, wenn keine Filterung der Ergebnisse durch einen Schwellenwert vorliegt. Dies ist z. B. sinnvoll, wenn ein guter Schwellenwert schwierig zu finden ist oder alle Modelle ausgegeben werden sollen. Letzterer Aspekt kann auch im Kontext der Wiederverwendung von Prozessmodellen durch eine Suchfunktion sinnvoll sein, wenn zu einer Anfrage alle Ergebnismodelle einer Modellbibliothek nach absteigender Ähnlichkeit sortiert aufgelistet werden sollen.

Tab. 5.9: Durchschnittliche R-Precision und Precision-at-5 Werte der drei Evaluationsdatensätze. Fett gedruckte Einträge kennzeichnen die besten Werte der entsprechenden Spalte.

	DM		**DM + UA + BR**		**DM + UA + BR + CM**	
	R-Prec.	Prec.-at-5	R-Prec.	Prec.-at-5	R-Prec.	Prec.-at-5
LS3-QueryAll	**0,98**	**0,99**	**0,96**	**0,99**	**0,96**	**0,99**
GEDS	0,97	**0,99**	0,89	0,95	0,78	0,91
FBSE	0,49	0,65	0,40	0,56	0,38	0,53
SSBOCAN	**0,98**	**0,99**	0,92	0,97	0,93	0,98
LAROSA	0,86	0,96	0,77	0,90	0,81	0,93

Die Ergebnisse aller drei in der Evaluation verwendeten Datensätze ergeben, dass der LS3-QueryAll Ansatz auch bezüglich R-Precision und Precision-at-5 besser abschneidet als die verglichenen Verfahren. Dabei sind die Werte durchgängig sehr hoch mit einem minimalen Wert von 0,96. Im Unterschied zu den zuvor beschriebenen Evaluationsergebnissen schneiden die SSBOCAN und GEDS Verfahren im Vergleich insbesondere für die Precision-at-5 Werte wesentlich besser ab. So liegt SSBOCAN maximal 0,02 Punkte zurück, während GEDS maximal 0,08 Punkte zurückliegt. Auch LAROSA erzielt gute Ergebnisse mit maximal 0,09 Punkten Rückstand auf den LS3-QueryAll Ansatz. FBSE schneidet dagegen deutlich schlechter mit einem maximalen Precision-at-5 Wert von 0,65 ab. Dies bedeutet, dass vor allem SSBOCAN beinahe gleich gute Ergebnisse erreicht wie die LS3-Verfahren, wenn nur die jeweils fünf besten Ergebnisse einer Anfrage berücksichtigt werden.

Werden hingegen die R-Precision Werte betrachtet, so fallen größere Differenzen zwischen dem LS3-Verfahren und den verglichenen Ansätzen auf. Ein vergleichbar hohes Niveau erzielen nun lediglich noch die Verfahren LS3-QueryAll und SSBOCAN mit Werten von mindestens 0,92. Der Abstand zu GEDS und LAROSA erhöht sich dagegen. Die beiden Ansätze liegen jetzt zwischen 0,07 und 0,18 Punkten zurück. Daraus folgt, dass die Matching-basierten Verfahren zwar durchaus ähnlich gute Werte für die fünf besten Ergebnisse berechnen, sich allerdings anschließend vermehrt inkorrekte Modelle in der Rangfolge befinden. Zudem lässt sich auch anhand dieser Kennzahlen wiederum erkennen, dass die Matching-basierten Verfahren bei schwieriger zu berechnenden Matches schlechtere Ergebnisse liefern, während der LS3-QueryAll Ansatz durchgängig sehr gute Ergebnisse erzielt. Im Vergleich zu den zuvor erläuterten F-Werten fällt der Rückgang der Ergebnisqualität allerdings nicht so stark aus.

5.4.3 Diskussion

Qualität der Ergebnisse für Precision, Recall und F-Wert: Bezüglich der zuvor präsentierten Ergebnisse lässt sich festhalten, dass die vier Ansätze LS3-QueryAll, LS3-Query, GEDS und SSBOCAN für die ersten beiden Stufen der Evaluation zufriedenstellende Resultate liefern, da sie jeweils einen F-Wert über 0,8 erzielen. Dabei sind insbesondere die Precision-Werte herauszuheben, deren kleinster Wert 0,93 beträgt. Und die Recall-Werte liegen ebenfalls nur geringfügig niedriger. Somit bedeuten diese Werte, dass die vier genannten Ansätze beinahe nur relevante Modelle für eine Anfrage zurückliefern (hohe Rate an True Positives), aber dass alle Verfahren auch einen

gewissen Teil der relevanten Modelle nicht als relevant einstufen (False Negatives). Dies ist letztlich auch der Grund, warum die F-Werte in der zweiten Evaluation nicht höher als 0,9 sind. Werden jedoch wie aus der dritten Evaluation ersichtlich mehr Modelle hinzugefügt, bei denen Matches schwierig zu berechnen sind, so erzielen nur noch die LS3-Verfahren F-Werte über 0,8. Daher kann aufgrund der vorliegenden Evaluation die Hypothese bestätigt werden, dass ähnlichkeitsbasierte Suchverfahren, denen Matches zugrunde liegen, Schwierigkeiten bei Modellen mit heterogenen Beschriftungen haben.

Dies lässt sich zudem anhand einer detaillierteren Betrachtung einzelner Anfragen nachvollziehen. So haben alle Verfahren bei der dritten Evaluationsstufe größere Probleme korrekte Ergebnisse für die BR- und UA-Datensätze zu finden als für den DM-Datensatz. Während die fünf besten Verfahren LS3-QueryAll, LS3-Query, LAROSA, GEDS und SSBOCAN zwar keine False Positives ermitteln, klassifizieren sie vergleichsweise viele relevante Modelle als nicht relevant, was zu einer großen Zahl an False Negatives führt. Dies trifft vor allem auf LAROSA, GEDS und SSBOCAN zu, während die LS3-Ansätze immerhin durchschnittliche Ergebnisse erzielen. Von den insgesamt 162 korrekten Modellen für die BR- und UA-Anfragen liefern LAROSA, GEDS und SSBOCAN nur 40 (25 %), 38 (24 %) bzw. 32 (20 %) Modelle zurück. Das LS3-QueryAll Verfahren ermittelt immerhin 76 (47 %) und der LS3-Query Ansatz noch 60 (37 %) korrekte Modelle. Für den DM-Datensatz kehren sich die Ergebnisse demgegenüber sogar teilweise um. LS3-QueryAll berechnet 758 (95 %) der 800 relevanten Modelle, LAROSA, SSBOCAN and GEDS liefern dagegen sogar 638 (80 %), 782 (98 %) bzw. 768 (96 %) der korrekten Modelle als Anfrageergebnisse.

Daraus ergibt sich, dass die LS3-Verfahren besser für Modellbibliotheken mit heterogenen Beschriftungen und sich unterscheidender Wortwahl geeignet erscheinen. Dies kann darauf zurückgeführt werden, dass Prozessmodell-Matching Ansätze Schwierigkeiten haben, für die BR- und UA-Datensätze korrekte Matches zu identifizieren (Antunes u. a. 2015; Cayoglu u. a. 2014). In der zuvor beschriebenen Evaluation waren beispielsweise LAROSA, GEDS und SSBOCAN nicht in der Lage genügend korrekte Matches zwischen den BR-Modellen zu identifizieren, sodass die berechneten Ähnlichkeitswerte zu gering waren, um die Modelle als ausreichend ähnlich im Sinne der Evaluation einzustufen. Selbige Schlussfolgerungen lassen sich auch hinsichtlich der CM-Modelle ziehen. So identifiziert beispielsweise der LS3-QueryAll Ansatz für die 40 Anfragen des CM-Datensatzes 382 von 400 korrekten Ergebnissen bei keinem False Positive (Precision-Wert von 1,0). Demgegenüber findet SSBOCAN nur 104 korrekte

Ergebnisse bei keinem False Positive und FBSE konnte zwar 232 korrekte Ergebnisse berechnen, allerdings wurden auch 839 False Positives ausgegeben. LAROSA konnte sich hingegen insgesamt aufgrund der CM-Modelle im Verhältnis zu den anderen Verfahren steigern, da immerhin 262 korrekte Ergebnisse bei keinem False Positive identifiziert wurden.

Die unterschiedlichen Werte zwischen den LS3-QueryAll und LS3-Query Verfahren ergeben sich aufgrund der notwendigen Berechnung von Pseudodokumenten gemäß Formel 5.6 im letzteren Fall. Die Projektion eines Anfragemodells in den semantischen Vektorraum erhöht der Evaluation zufolge zwar die Precision-Werte, verringert jedoch in größerem Maße den Recall. Der LS3-Query Ansatz weist vor allem für einen Prozess des DM-Datensatzes sehr geringe durchschnittliche Recall- und F-Werte von 0,31 bzw. 0,45 auf. Der LS3-Query Ansatz verringert allerdings dennoch in der zweiten Stufe der Evaluation gegenüber der ersten Stufe den Abstand der F-Werte zu den GEDS und SSBOCAN Verfahren. Von der ursprünglichen Differenz von 0,1 (Macro-Average) bzw. 0,06 (Micro-Average) Punkten reduziert sich die Differenz auf 0,02 bzw. 0,01 Punkte im Macro-Average Fall und jeweils 0,03 Punkte im Micro-Average Fall. In der dritten Stufe der Evaluation schneidet der LS3-Query Ansatz schließlich sogar besser ab als GEDS und SSBOCAN.

Zusammenfassend lässt sich hinsichtlich der Eignung der verschiedenen Verfahren für eine ähnlichkeitsbasierte Suche feststellen, dass die Qualität der LS3-Verfahren etwas unter der anderer Matching-basierter Ansätze liegt, solange Matches einfach zu berechnen sind. Dies ist etwa dann der Fall, wenn die Modelle durch Process Minig Techniken automatisch aus Prozessausführungen generiert werden, da dann eine einheitliche Beschriftung gewährleistet werden kann. Sollten jedoch Matches schwierig zu berechnen sein, was typischerweise der Fall ist, wenn die Modelle von Menschen erstellt werden, liegen die LS3-Verfahren qualitativ deutlich über den anderen verglichenen Ansätzen.

Qualität der Ergebnisse für Precision-at-5 und R-Precision: Bezüglich der Precision-at-5 Werte liefern alle Verfahren mit Ausnahme von FBSE gute bis sehr gute Ergebnisse. Gleiches gilt prinzipiell auch für die R-Precision Werte, wobei allerdings die Abstände zwischen LS3-QueryAll und den verglichenen Verfahren gegenüber den Precision-at-5 Werten größer werden. Der Unterschied von 0,18 Punkten zwischen LS3-QueryAll und GEDS in der dritten Evaluationsstufe ist bereits beträchtlich. Dabei ist noch bemerkenswert, dass mit LS3-QueryAll und GEDS die beiden Verfahren am

besten abschneiden, die lediglich die textuellen Beschriftungen der Modelle berücksichtigen. Für die R-Precision erzielen sie zudem noch deutlich bessere Ergebnisse als die verglichenen Verfahren, die auch die Struktur- bzw. Verhaltensdimension miteinbeziehen. Es erscheint daher so, als ob diese zusätzlichen Informationen bei der ähnlichkeitsbasierten Suche keinen Vorteil einbringen. Dies liegt vermutlich wiederum an der Schwierigkeit korrekte Matches zu identifizieren. Die Berechnung der Graph-Editier Distanz von GEDS beispielsweise verschlechtert einen berechneten Ähnlichkeitswert potentiell mehrfach bei nicht gefundenen korrekten Matches. So wird der Ähnlichkeitswert zum einen verringert, da Matches nicht identifiziert wurden; zum anderen wird er reduziert, da damit zusammenhängend Kanten ebenfalls falsch interpretiert werden. Um abschließend zu klären, ob Struktur- oder Verhaltensdimensionen nützlich für eine ähnlichkeitsbasierte Suche sind, könnten die Suchergebnisse basierend auf einem korrekten Matching berechnet werden und die resultierenden Anfrageergebnisse mit den LS3-Ergebnissen verglichen werden.

Das schlechtere Abschneiden von LAROSA, GEDS und SSBOCAN in Bezug auf R-Precision lässt sich ebenfalls auf die UA- und BR-Datensätze zurückführen. Während LS3-QueryAll 130 (80 %) der 162 korrekten Modelle ermittelt, erkennen SSBOCAN lediglich 111 (68 %), GEDS 83 (51 %) und LAROSA 63 (39 %) korrekte Ergebnisse. Hinsichtlich des DM-Datensatzes ergeben sich hingegen bessere Werte für LAROSA, GEDS und SSBOCAN. SSBOCAN berechnet 781 (98 %), GEDS 779 (97 %) und LAROSA 679 (85 %) der 800 korrekten Modelle. LS3-QueryAll schneidet mit 782 (98 %) korrekten Modellen kaum besser ab.

Laufzeit und Komplexität: Ein weiterer generell großer Unterschied zwischen den LS3-Verfahren und allen anderen Ansätzen betrifft die gemessene Laufzeit. Zumindest mit den zur Verfügung stehenden Implementierungen erscheinen nur die LS3-Ansätze für den praktischen Einsatz in einer Suchfunktion geeignet zu sein. Die Laufzeitunterschiede treten auf, da die LS3-Verfahren kein aufwändiges Prozessmodell-Matching durchführen müssen, um Ähnlichkeitswerte zu berechnen wie im Falle der verglichenen Ansätze. In den LS3-Algorithmen müssen die Modelle lediglich ein Mal ausgelesen werden, um aus ihnen Dokumentvektoren zu erzeugen und daraus eine Term-Dokument Matrix zu erstellen.

Der Einsatz von Matching-basierten Suchverfahren könnte somit aufgrund der hohen Laufzeit selbst bei einfach zu berechnenden Matches wenig hilfreich sein. Zumindest unter der Annahme, dass Nutzer einer solchen Suchfunktion nicht Minuten

oder Stunden auf ein Ergebnis warten können, erscheint trotz der leicht besseren Ergebnisse der GEDS und SSBOCAN Verfahren gegenüber den LS3-Ansätzen für die erste Evaluationsstufe die wesentlich längere Berechnungszeit nicht angemessen.

Bezüglich der theoretischen Komplexität hängt die Abschätzung für die LS3-Algorithmen von der Anzahl an Termen in den Prozessmodellen $|W_{All}|$ sowie von der Anzahl an Modellen n ab. Die folgende Betrachtung gilt für den LS3-QueryAll Algorithmus und berücksichtigt die einzelnen Schritte des Verfahrens:

- **Extraktion von Termen:** Die Komplexität dieses Schritts wird zum einen von der Anzahl an Termen eines Modells ($|w(m)|$) und zum anderen von der Anzahl an Modellen n beeinflusst. Aus jedem Modell müssen die Terme extrahiert sowie die Vorverarbeitungsschritte durchgeführt werden, sodass sich eine Komplexität von $\mathcal{O}(|w(m)| \cdot n)$ ergibt, um eine Term-Dokument Matrix zu erzeugen.
- **Gewichtung der Term-Dokument Matrix:** Bei der Gewichtung der Term-Dokument Matrix müssen alle Einträge neu berechnet werden. Wenn die Term-Dokument Matrix $|W_{All}|$ Zeilen und n Spalten (die Anzahl an Modellen) aufweist, so beträgt die Komplexität für diesen Schritt $\mathcal{O}(|W_{All}| \cdot n)$.
- **Singulärwertzerlegung:** Für die Singulärwertzerlegung beträgt die Komplexität $\mathcal{O}(min(|W_{All}|^2 \cdot n, |W_{All}| \cdot n^2))$ mit $|W_{All}|$ der Anzahl an Zeilen und n der Anzahl an Spalten (siehe auch (Holmes u. a. 2008)).
- **Berechnung von Ähnlichkeitswerten:** Zur Berechnung der Ähnlichkeitswerte sind $\frac{n \cdot (n+1)}{2}$ Berechnungen der Kosinus-Ähnlichkeit notwendig. Es müssen dabei nicht für alle Kombinationen von Modellen Ähnlichkeitswerte berechnet werden, da das LSSM Ähnlichkeitsmaß symmetrisch ist (siehe Diskussionspunkt Metrik-Eigenschaften). Daher liegt die Komplexität der Ähnlichkeitsberechnung in der Klasse $\mathcal{O}(n^2)$.
- **Berechnung von Anfrageergebnissen:** Um die letztendlichen Anfrageergebnisse zu bestimmen, sind schließlich noch $\frac{n \cdot (n+1)}{2}$ Vergleiche von Einträgen einer LSSM-Matrix mit einem Schwellenwert θ notwendig. Somit ergibt sich analog zur Berechnung von Ähnlichkeitswerten $\mathcal{O}(n^2)$.

Entsprechend diesen Punkten liegt die Komplexität von LS3-QueryAll in der Klasse $\mathcal{O}(min(|W_{All}|^2 \cdot n, |W_{All}| \cdot n^2))$. Dabei ist davon auszugehen, dass typischerweise mehr Terme in W_{All} als Dokumente in einer Modellbibliothek vorhanden sind, was in einer

Komplexität von $\mathcal{O}(|W_{All}|\cdot n^2)$ resultiert. Diese hohe Komplexität aufgrund der Singulärwertzerlegung ist für Anwendungen mit sehr großen Datensätzen nicht geeignet, es wurden jedoch zum einen Algorithmen mit geringerer Laufzeit unter Einbeziehung einer Fehlerrate vorgeschlagen (Holmes u. a. 2008). Zum anderen muss die Singulärwertzerlegung nur neu berechnet werden, wenn sich der Modellbestand ändert und nicht für jede Anfrage (siehe dazu auch die Ausführungen zu den Algorithmen in Kapitel 5.3).

Für den LS3-Query Algorithmus reduziert sich die Komplexitätsklasse, wenn davon ausgegangen wird, dass die Singulärwertzerlegung und Dimensionsreduktion bereits durchgeführt wurden. In diesem Fall muss zunächst für ein Anfragemodell ein Pseudo-Dokument erzeugt werden, was aufgrund der Vektor- und Matrixmultiplikation in der Klasse $\mathcal{O}(2 \cdot |W_{All}| \cdot k^2)$ liegt (k entspricht der Anzahl an Dimensionen der Dimensionsreduktion und $|W_{All}|$ der Anzahl an Termen). Anschließend müssen die Ähnlichkeitswerte eines Anfragemodells zu den Modellen einer Modellbibliothek berechnet werden, wofür n Berechnungen der Kosinus-Ähnlichkeit notwendig sind ($\mathcal{O}(n)$). Abschließend werden noch n Vergleiche der berechneten Ähnlichkeitswerte mit einem Schwellenwert θ benötigt, sodass die Gesamtkomplexität von LS3-Query in $\mathcal{O}(2 \cdot |W_{All}| \cdot k^2)$ liegt, da die Anzahl an Termen $|W_{All}|$ die Anzahl an Dokumenten n übersteigt.

Ähnlichkeits- und Schwellenwerte: Ein weiterer Aspekt, der bei genauerer Betrachtung der von den verschiedenen Verfahren berechneten Ähnlichkeitswerte auffällt, betrifft die unterschiedlichen Schwellenwerte. Da für alle Verfahren die Ähnlichkeitswerte auf das Intervall $[0, 1]$ normiert wurden, ist intuitiv zu erwarten, dass ähnliche Modelle einen Wert nahe 1 erhalten. Im Kontext der Evaluation sollten also die Modelle, die den gleichen zugrundeliegenden Prozess darstellen, hohe Werte erhalten, wohingegen für die anderen Modelle eher niedrige Werte vermutet werden.

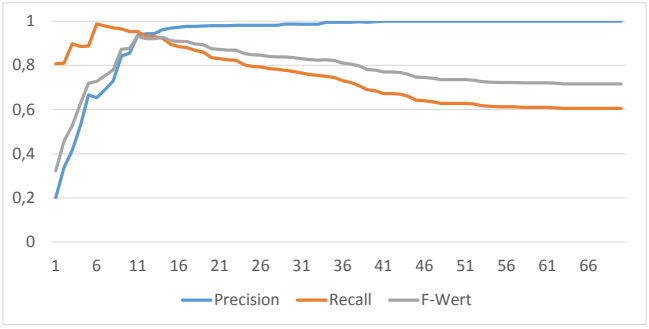
Bei einer Untersuchung der Ähnlichkeitswerte fällt auf, dass insbesondere das LARO-SA Verfahren im Allgemeinen niedrige Werte berechnet, sodass der beste Schwellenwert für die verschiedenen Evaluationsfälle bei 0,18 bzw. 0,19 liegt. Dies widerspricht der intuitiven Vorstellung der Verteilung der Ähnlichkeitswerte und könnte möglicherweise zu einer geringeren Qualität der Suchergebnisse führen, wenn noch mehr Modelle in einer Bibliothek enthalten sind, da sich True Positives schwieriger von False Positives unterscheiden lassen.

Bei FBSE zeigt sich das gegenteilige Bild. Bei diesem Verfahren werden im Allgemeinen sehr hohe Ähnlichkeitswerte berechnet, sodass sich zwar ein intuitiv sinnvoller hoher Schwellenwert festlegen lässt (in der Evaluation 0,88 bzw. 0,92). Allerdings tritt dabei schon bei der Evaluation das für das LAROSA-Verfahren skizzierte Problem der Unterscheidbarkeit von True Positives und False Positives auf. Für die dritte Evaluationsstufe finden sich mit einem Schwellenwert von 0,92 immer noch 2.346 False Positives in den Anfrageergebnissen. Bei LAROSA mit dem zweithöchsten Wert sind es hingegen nur noch 207 und bei LS3-Query gar nur 20 als geringster Wert.

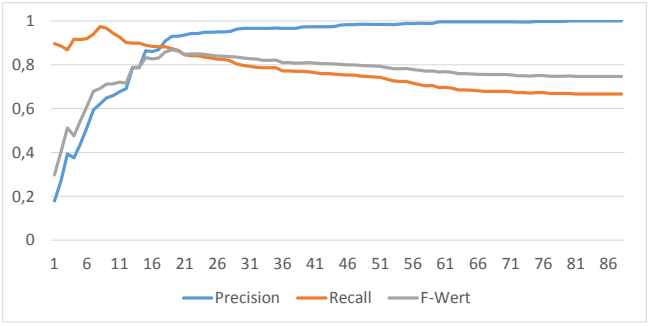
Hier scheinen die LS3-Verfahren die Ähnlichkeitswerte im Intervall $[0, 1]$ sinnvoll zu verteilen. Der intuitiven Auffassung folgend, kann bei diesen Verfahren ein relativ hoher Schwellenwert gewählt werden – bei der Evaluation ein Wert im Intervall $[0,76, 0,8]$ –, ohne dabei viele False Positives oder False Negatives zu erzeugen.

Dimensionalität k: Allerdings weisen die LS3-Suchverfahren auch Schwächen auf. Da sie auf der LSA basieren, haben sie die gleichen Nachteile wie die LSA. Insbesondere kann es schwierig sein den besten Wert für den Dimensionalitätsparameter k zu finden, da dieser in einer großen Menge sinnvoller Werte liegen kann. Wie bereits in Kapitel 3.3 im Kontext der LSA erwähnt, ergibt sich generell immer als sinnvoller Wertebereich das Intervall zwischen 1 und dem Rang der Singulärwertzerlegung. Die Dimensionalität 1 zu wählen ergibt jedoch wenig Sinn, da dann nur noch eine Unterscheidung zwischen ähnlich (Wert 1) und nicht ähnlich (Wert 0) getroffen werden kann. Den Rang der Singulärwertzerlegung als Dimensionalität zu wählen ergibt ebenso typischerweise nicht die besten Ergebnisse. So wurde bereits in (Deerwester u. a. 1990) im Information Retrieval Kontext festgestellt, dass die Verwendung einer hohen Dimensionsanzahl Suchergebnisse nicht notwendigerweise verbessert. Dies ist auch in Abbildung 5.5 für die Evaluationsdatensätze zu erkennen. Daher liegt ein sinnvoller Wert zwischen den beiden Intervallgrenzen.

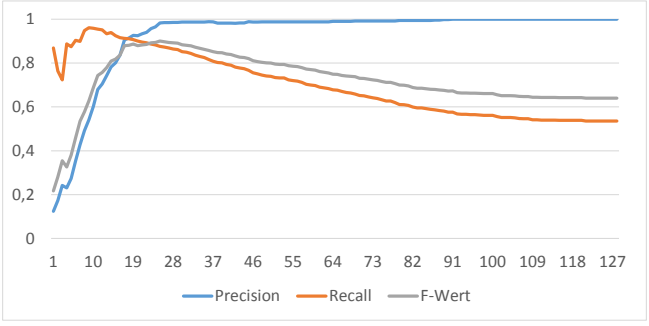
Für die drei in der Evaluation verwendeten Modellbibliotheken sind das die Mengen $\{1, ..., 69\}$, $\{1, ..., 88\}$ und $\{1, ..., 128\}$. Während sich zwar für die meisten Werte von k immer noch ein relativ guter F-Wert ergibt (siehe Abbildung 5.5), sinkt er bei zu geringen Werten von k deutlich ab. Wird beispielsweise für die zweite Evaluationsstufe $k = 6$ gesetzt, würde dies einen F-Wert von 0,61 ergeben. Identisch ist jedoch stets der generelle Verlauf der Precision, Recall und F-Wert Kurven in Abhängigkeit von k: Mit größer werdendem k steigen die Precision-Werte zum optimalen Wert 1 hin an, während sich Recall und F-Wert nach einem Hochpunkt zunehmend verringern.



(a) Datensatz mit DM-Modellen ($k_{max} = 69$)



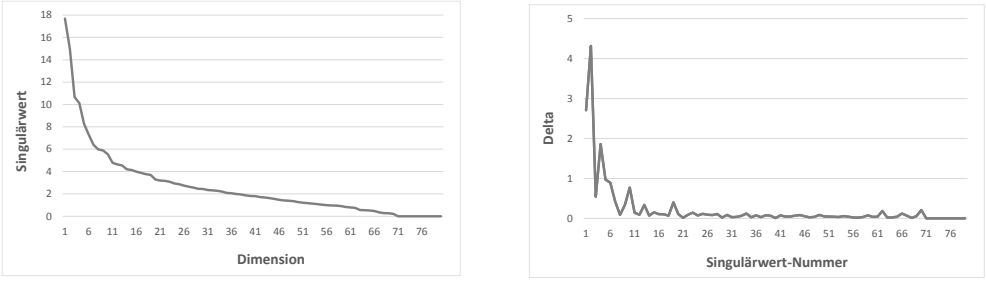
(b) Datensatz mit DM, BR und UA-Modellen ($k_{max} = 88$)



(c) Datensatz mit DM, BR, UA und CM-Modellen ($k_{max} = 128$)

Abb. 5.5: Verlauf der Werte von Precision, Recall und F-Wert für verschiedene Werte von k für die in der Evaluation verwendeten Datensätze. Die x-Achse zeigt jeweils die Dimensionalität, während auf der y-Achse der Wert der Kennzahlen abgebildet ist.

Eine mögliche Heuristik zur Identifikation eines optimalen Werts für k könnte in der Betrachtung der Singulärwerte liegen. Da durch die Multiplikation der Singulärwert-Matrix Σ und der Dokumentmatrix D^T eine Skalierung bzw. Gewichtung von D^T stattfindet,[7] könnte als Indikator für wenig einflussreiche Dimensionen die Höhe der Singulärwerte betrachtet werden. Sollte sich demzufolge die Höhe von aufeinander folgenden Singulärwerten nur noch wenig unterscheiden, kann dies als Indiz gewertet werden, dass die weiteren Dimensionen nur noch eine geringe Rolle spielen und damit die optimale Dimensionalität gefunden wurde. Abbildung 5.6b zeigt exemplarisch für die DM-Modelle die Differenz von zwei aufeinander folgenden Singulärwerten. Wie zu sehen ist, wird die Differenz zwischen zwei Werten bis auf kleine Ausreißer beständig geringer. Für Werte von k größer als dem optimalen Wert 14 ergeben sich nur noch geringe Unterschiede, sodass dies als Hinweis für einen Bereich mit einem guten Wert für k dienen kann. Zugleich ist in Abbildung 5.6a zu erkennen, dass auch aufgrund der absoluten Größe der Singulärwerte Dimensionen größer 14 eine zunehmend geringere Rolle spielen gegenüber Dimensionen kleiner 14. Dieser grundsätzliche Verlauf der beiden Grafiken ist auch für die beiden anderen Evaluationsstufen ersichtlich (siehe Anhang A.2.2).



(a) Singulärwerte **(b)** Differenz der Singulärwerte

Abb. 5.6: Singulärwerte und Differenz von zwei aufeinander folgenden Singulärwerten für die erste Evaluationsstufe (DM-Modelle)

Daraus ergibt sich zusammenfassend, dass die Wahl der Dimensionalität k die Qualität der Suchergebnisse beeinflusst und die Wahl einer guten Dimensionsanzahl aufgrund der zumeist großen Anzahl an Dimensionen schwierig ist. Als Heuristik zur

[7] Die Singulärwert-Matrix ist nach dem höchsten Singulärwert absteigend sortiert, sodass die latenten Dimensionen der Matrix D^T entsprechend mehr oder weniger stark skaliert werden.

Lösung dieser Problematik wird die Betrachtung der Höhe und Differenzen zwischen aufeinander folgenden Singulärwerten vorgeschlagen, um zumindest einen Bereich für k einzugrenzen. Zum einen sollten Dimensionen mit im Verhältnis aller Singulärwerte großen Werten auf jeden Fall zum Aufspannen des semantischen Suchraums verwendet werden (untere Grenze). Zum anderen können kleine Unterschiede aufeinander folgender Singulärwerte als gute obere Grenze angesehen werden. Alternativ dazu kann die Dimensionsanzahl, potentiell unter Berücksichtigung der zuvor bestimmten Intervallgrenzen, auch über statistische Modellvalidierungstechniken wie z. B. Kreuzvalidierung bestimmt werden (Kohavi 1995).

Einfluss von Stoppwörtern und Wortstammreduktion: Tabelle 5.10 stellt den Einfluss auf die Ergebnisse durch die Entfernung von Stoppwörtern und die Wortstammreduktion für Precision, Recall und F-Wert dar. Es zeigt sich eine relativ deutliche Verschlechterung dieser Werte, wenn keine Wortstammreduktion durchgeführt wird bzw. wenn Stoppwörter nicht entfernt werden. Dies gilt für alle drei in der Evaluation verwendeten Datensätze.

Tab. 5.10: Einfluss von Wortstammreduktion und Entfernung von Stoppwörtern auf Precision, Recall und F-Wert. In der ersten Spalte sind die verwendeten Datensätze und in der zweiten Spalte die jeweils nicht berücksichtigten Techniken angegeben.

Modelle	**Entfernte Technik**	**LS3-QueryAll**		
		Precision	Recall	F-Wert
DM	Stoppwort	0,82	0,92	0,87
	Wortstammreduktion	0,86	0,86	0,86
	Stoppwort + Wortstammreduktion	0,78	0,87	0,82
	Vergleichswert	0,95	0,92	0,93
DM + UA + BR	Stoppwort	0,82	0,85	0,83
	Wortstammreduktion	0,87	0,79	0,83
	Stoppwort + Wortstammreduktion	0,80	0,80	0,80
	Vergleichswert	0,96	0,85	0,90
DM + UA + BR + CM	Stoppwort	0,89	0,88	0,89
	Wortstammreduktion	0,90	0,86	0,88
	Stoppwort + Wortstammreduktion	0,87	0,86	0,86
	Vergleichswert	0,97	0,88	0,93

Aus den Werten in Tabelle 5.10 zeigt sich, dass die Precision-Werte im Vergleich zu den ursprünglichen Werten geringer ausfallen, wenn auf die Entfernung von Stoppwörtern verzichtet wird. Wird hingegen keine Wortstammreduktion angewendet, so fallen sowohl Precision- als auch Recall-Werte niedriger aus. Daraus folgend fallen diese Werte auch geringer aus, wenn sowohl auf die Entfernung von Stoppwörtern als auch auf die Wortstammreduktion verzichtet wird. Für die Evaluationsdatensätze ergibt sich ein um ungefähr 0,1 geringerer F-Wert, der die Wichtigkeit dieser Techniken für das LS3-Verfahren verdeutlicht.

Anzahl an Wörtern und Wortwahl: Die LS3-Suchverfahren werden sehr gut funktionieren, wenn die Modelle eine ausreichende Anzahl an Wörtern enthalten und die Wortwahl zwischen Modellen, die als unähnlich zu klassifizieren sind, möglichst stark differiert. Dann unterscheiden sich die Dokumentenvektoren unähnlicher Modelle hinsichtlich ihrer Richtung innerhalb des semantischen Vektorraums der LSA deutlich voneinander. Wie sich anhand von Tabelle 5.3 erkennen lässt, sind die Unterschiede der einzelnen Datensätze der Evaluation bezüglich der Anzahl an Modellen und Termen recht deutlich. Die differierenden Ergebnisse der LS3-Verfahren für die verschiedenen Datensätze lassen sich anhand dieser Charakteristiken erklären.

Die UA- und BR-Modelle stellen eine Herausforderung für die LS3-Verfahren dar, obwohl sie recht viele Terme enthalten (149 bzw. 141), da diese sich jedoch stark von Modell zu Modell unterscheiden. Damit ist die Wortwahl innerhalb dieser Modelldatensätze zu verschieden, sodass sich zu niedrige Ähnlichkeitswerte ergeben. Die DM- und CM-Modelle enthalten hingegen bezogen auf die Modellanzahl relativ wenige eindeutige Terme (391 bzw. 191) und die Wortwahl unterscheidet sich kaum für Modelle, die den gleichen Prozess repräsentieren. Somit sind bei der Evaluation des CM-Datensatzes auch relativ wenige Terme ausreichend, um sehr gute Anfrageergebnisse zu berechnen. Die CM-Modelle enthalten durchschnittlich nur 36,25 Terme, dennoch werden 382 der 400 korrekten Modelle ermittelt, wobei kein False Positive auftritt.

Die insgesamt 14 Prozesse der vier Modelldatensätze weisen aufgrund der unterschiedlichen thematischen Ausrichtung auch untereinander wenig übereinstimmende Terme auf. Daher sind die LS3-Verfahren sehr gut in der Lage die Modelle der verschiedenen Prozesse voneinander zu separieren. In diesem Zusammenhang wäre noch eine weitere Untersuchung interessant, bei der für eine größere Menge an Modellen eines Prozesses die Anfrageergebnisse bewertet werden. Beispielsweise wenn

für den Bewerbungsprozess an Universitäten 100 Modelle vorhanden wären und eine durch Menschen bestimmte Rangfolge mit den LS3-Ergebnissen verglichen wird.

Auswahl von LSA: Anstatt die Latent Semantic Analysis zu verwenden, könnten auch anderen Verfahren aus dem Bereich der *Topic Models* – wie beispielsweise die *Latent Dirichlet Allocation* (LDA) (Blei u. a. 2003) – bzw. Algorithmen, die sogenannte *Word Embeddings* erzeugen – wie z. B. *Word2vec* (Mikolov u. a. 2013b) –, eingesetzt werden. Diesen Verfahren wird in Untersuchungen aus dem Bereich der Computerlinguistik eine höhere Qualität gegenüber der LSA eingeräumt (Baroni u. a. 2014), allerdings zeigt beispielsweise (Kusner u. a. 2015), dass die LSA auch besser abschneiden kann als die LDA. Zudem erscheint der Einsatz der LDA für die Ähnlichkeitsberechnung von Prozessmodellen in (Qiao u. a. 2011) mit den vergleichsweise geringen beschriebenen Precision Werten nicht erfolgreich zu sein. Die Precision Werte reichen von 0,4 bis 0,79 für verschiedene Datensätze. Für zukünftige Forschungsarbeiten zur ähnlichkeitsbasierten Suche nach Prozessmodellen könnten Ansätze wie LDA oder WORD2VEC bei vorhandenen Trainingsdaten und entsprechender Paramterwahl jedoch durchaus bessere Ergebnisse erzielen als die LS3.

Neben der zuvor beschriebenen Auswahl der LSA könnte auch ein klassischer *Word-Matching* Ansatz aus dem Information Retrieval verwendet werden, bei dem die Kosinus-Ähnlichkeit nach dem 2. Schritt basierend auf einem Vektorraum der gewichteten Term-Dokument Matrix berechnet wird (siehe z. B. (Dumais 2007)). Der Begriff Word-Matching bezieht sich dabei auf den rein syntaktischen Vergleich von Termen in Modellen. Der Vektorraum wird beim Word-Matching aus der Termmenge W_{all} und nicht aus den latenten semantischen Dimensionen gebildet. Wie Abbildung 5.7 zeigt, schneidet das LS3-QueryAll Verfahren jedoch hinsichtlich der meisten Dimensionen besser oder gleich gut ab wie der Word-Matching Ansatz bei Betrachtung des F-Werts. Für die Analyse wurden die DM-Modelle gewählt, da sie im Vergleich zu den anderen Modelldatensätzen die höchste Übereinstimmung an Termen innerhalb der Modelle aufweisen, die den gleichen Prozess repräsentieren. Somit ergeben sich für diesen Modelldatensatz die besten Werte für das Word-Matching Verfahren. Zusammenfassend zeigt sich hierbei, dass die LS3-Ansätze einem klassischen Word-Matching Verfahren überlegen sind (für weitere Details siehe (Schoknecht u. a. 2017a)).

Metrik-Eigenschaften: Bezüglich der Metrik-Eigenschaften muss die Ähnlichkeitsberechnung des LSSM in ein Distanzmaß überführt werden, um die Bedingungen

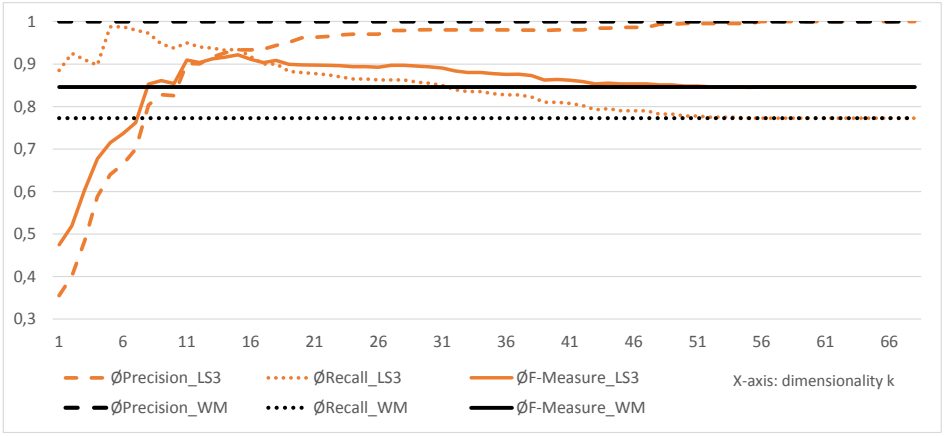


Abb. 5.7: Vergleich des LS3-QueryAll Ansatzes mit einem klassischen Word-Matching Verfahren aus dem Information Retrieval (Quelle: (Schoknecht u. a. 2017a)). Der zugrunde liegende Datensatz sind die DM-Modelle.

aus Definition 2.8 überprüfen zu können. Eine einfache Variante wäre die Berechnung der Distanz $dist$ zweier Prozessmodelle als $dist = 1 - LSSM(m_x, m_y)$. Damit ergeben sich Distanzwerte im Intervall $[0, 1]$, wobei sich die minimale Distanz von 0 bei der maximalen Ähnlichkeit von 1 ergibt und umgekehrt der höchste Distanzwert von 1 bei der minimalen Ähnlichkeit von 0 erzielt wird. Da die für die LS3-Verfahren verwendete Kosinus-Ähnlichkeit die Dreiecksungleichheit bei dieser Variante jedoch nicht erfüllt (van Dongen und Enright 2012), wird auf die folgende, ebenso in (van Dongen und Enright 2012) vorgeschlagene Variante, zurückgegriffen: $dist = \sqrt{1 - LSSM(m_x, m_y)^2}$. Diese Transformation erfüllt die vier Bedingungen einer Metrik: Nichtnegativität, Symmetrie, Identität und Dreiecksungleichheit.

Nichtnegativität: Die Distanzfunktion ist immer größer oder gleich 0, womit diese Eigenschaft erfüllt ist. Damit $dist = \sqrt{1 - LSSM(m_x, m_y)^2}$ negative Werte annehmen kann, müsste $LSSM(m_x, m_y) > 1$ sein. Da die Werte der Ähnlichkeitsfunktion $LSSM(m_x, m_y)$ im Intervall $[0, 1]$ liegen, ist auch der maximale Wert von $LSSM^2 = 1$, sodass der minimale Wert von $dist = 0$ ist.

Symmetrie: Die Distanzfunktion $dist = \sqrt{1 - LSSM(m_x, m_y)^2}$ kann nur die Symmetrie-Eigenschaft verletzten, wenn $LSSM(m_x, m_y) \neq LSSM(m_y, m_x)$ ist. Da jedoch jeweils aus m_x und m_y der gleiche Dokumentenvektor erzeugt wird, ergibt sich der-

selbe Winkel zwischen diesen Vektoren. Damit ist ebenso der Wert der Kosinus-Ähnlichkeit sowie daraus folgend der Distanzwert gleich. Somit erfüllt die Distanzfunktion $dist$ auch die Symmetrie-Eigenschaft.

Identität: Zur Überprüfung der Identitätseigenschaft können die beiden Eigenschaften *Reflexivität* und *Positivität* herangezogen werden (Zezula u. a. 2006, S. 8). Reflexivität ist dabei als $\forall x \in \mathcal{D}, d(x,x) = 0$ und Positivität als $\forall x, y \in \mathcal{D}, x \neq y \Rightarrow d(x,y) > 0$ definiert. Die Distanzfunktion $dist = \sqrt{1 - LSSM(m_x, m_y)^2}$ erfüllt formal auch diese Eigenschaften, da hinsichtlich der Reflexivität ein Modell m_x jeweils auf den gleichen Vektor abgebildet wird und demzufolge die Kosinus-Ähnlichkeit 1 beträgt, was zu einer Distanz von 0 führt. Die Eigenschaft Positivität könnte zwar theoretisch verletzt werden, wenn zwei Modelle m_x und m_y genau die gleichen Terme für die Ähnlichkeitsberechnung enthalten. Diese Bedingung erscheint allerdings in der Praxis sehr unrealistisch, wenn bedacht wird, dass dann mehrere Modellierer genau die gleichen Beschriftungen wählen müssten bzw. ein Modellierer mehrfach das gleiche Modell erstellen müsste. Sollten jedoch tatsächlich zwei Modelle diese Bedingung erfüllen, können sie auch als das gleiche Modell aufgefasst werden, sodass die Eigenschaft Positivität ebenfalls erfüllt ist.

Dabei ist allerdings noch zu erwähnen, dass die Ähnlichkeitsberechnung mit der LSSM-Funktion lediglich die Beschriftungen von Modellelementen berücksichtigt. Andere Dimensionen wie das Ablaufverhalten oder die Graphstruktur (siehe auch Kapitel 2.2.2) werden hingegen nicht berücksichtigt. So könnte es theoretisch vorkommen, dass wie in Abbildung 2.5 die Modelle „Verkäuferprozess 1“ und „Verkäuferprozess 2“ die gleichen Beschriftungen enthalten, sie sich jedoch in ihrem Ablaufverhalten unterscheiden und damit nicht hinsichtlich aller Dimensionen exakt übereinstimmen, wie die Identitätseigenschaft suggeriert. Da die beschriebenen Suchfunktionen auf die natürlich-sprachliche Dimension fokussieren, ohne weitere Dimensionen zu berücksichtigen, wird nur diese als relevant betrachtet und die Distanzfunktion $dist$ erfüllt demzufolge auch die Identitätseigenschaft.

Dreiecksungleichheit: Zudem ist aus den Erläuterungen in (van Dongen und Enright 2012) ersichtlich, dass die Distanzfunktion $dist = \sqrt{1 - LSSM(m_x, m_y)^2}$ auch die Eigenschaft der Dreiecksungleichheit erfüllt. Somit handelt es sich bei der angegebenen Distanzfunktion um eine Metrik und insbesondere die Eigenschaften Symmetrie, Nichtnegativität und Dreiecksungleichheit können in angepassten Algorithmen dazu

verwendet werden die Berechnungszeit von Suchergebnissen weiter zu reduzieren (für eine Übersicht möglicher Ansätze siehe (Zezula u. a. 2006, Kap. 7)).

Weitere Anwendungsfälle: Für andere Anwendungsfälle sind die LS3-Verfahren sowie die Ähnlichkeitsberechnung durch den LSSM nicht geeignet, wenn einzelne Matches benötigt werden, da diese nicht berechnet werden. Dies ist beispielsweise für Einsatzzwecke aus dem Bereich des Conformance Checking[8] der Fall. Zu dieser Kategorie zählt etwa die Überprüfung von Prozessmodellen auf Entsprechungen zu einem Referenzmodell. Dabei sollen typischerweise zusätzliche, fehlende oder abweichende Aktivitäten bzw. unterschiedliche Reihenfolgen von Aktivitäten entdeckt werden. Hierzu reicht ein bloßer Ähnlichkeitswert nicht aus, sondern es müssen detailliert sich entsprechende Aktivitäten identifiziert werden, was in der zwingenden Berechnung von Matches resultiert. Ein weiterer Anwendungsfall für den die LS3-Ansätze nicht geeignet sind, ist das Auffinden von identischen bzw. sehr ähnlichen Modellteilen, sodass diese – möglichst sogar automatisiert – in ein eigenes Modell ausgelagert werden können (siehe dazu z. B. (La Rosa u. a. 2015)). Auch in diesem Fall müssen Matches berechnet werden, um sich entsprechende Teilmodelle aufzufinden. Des Weiteren muss zur Erkennung von auslagerbaren Modellteilen auch die Graphstruktur eines Prozessmodells berücksichtigt werden, die in der jetzigen Form nicht von den LS3-Verfahren verglichen wird.

Der Einsatz der LS3-Verfahren als eine Art Vorverarbeitungsschritt zum Auffinden ähnlicher Modelle wäre allerdings denkbar. So könnten zunächst die zueinander ähnlichen Modelle identifiziert werden, um anschließend mit einem anderen Verfahren Matches zwischen den Modellen für den spezifischen Einsatzzweck zu berechnen. Daraus würde sich der Vorteil ergeben, dass die aufwendige Berechnung von Matches nur noch zwischen den aus der LS3-Berechnung resultierenden ähnlichen Modellen durchgeführt werden müsste. Somit würde der Zeitbedarf aufgrund der geringen Laufzeit der LS3-Verfahren sinken. Für den Einsatzzweck der Überprüfung von Prozessmodellen gegenüber Referenzmodellen könnten etwa die Referenzmodelle als Anfragemodelle verwendet werden, um mit den LS3-Ansätzen ähnliche Modelle zu identifizieren. Daraufhin könnte ein spezifisches Verfahren zum Vergleich der Referenzmodelle mit den ähnlichen Modellen eingesetzt werden, sodass nicht alle Modelle einer Modellbibliothek mit den Referenzmodellen verglichen werden müssten.

[8] Weitere Details zu den Einsatzszenarien von Ähnlichkeitsberechnungen zwischen Prozessmodellen sind in der Übersicht in Kapitel 2.2.3 beschrieben.

5.5 Zusammenfassung

In diesem Kapitel wurde durch die *Latent Semantic Analysis-based Similarity Search* (LS3) Verfahren eine Möglichkeit beschrieben, um in einer Modellbibliothek zu einem Anfragemodell ähnliche Modelle zu finden. Diese Verfahren basieren auf der *Latent Semantic Analysis* (siehe auch Kapitel 3.3), mit deren Hilfe die Ähnlichkeit von Prozessmodellen bestimmt wird. Dazu werden aus Modellen Dokumentvektoren erzeugt, die als Vektoreinträge die Anzahl der unterschiedlichen Wörter eines Modells enthalten. Die Ähnlichkeit von zwei Modellen wird schließlich über die Kosinus-Ähnlichkeit der entsprechenden Vektoren berechnet. Durch diese Art der Ähnlichkeitsberechnung wird insbesondere die Notwendigkeit zur Bestimmung von Matches zwischen Prozessmodellen vermieden, die sich als schwierig erwiesen hat (siehe Kapitel 4 sowie (Antunes u. a. 2015; Cayoglu u. a. 2014)) und die für die meisten anderen Ansätze zur Ähnlichkeitsberechnung notwendig ist. Neben zwei Suchalgorithmen wurden auch Verfahren beschrieben, um mit Änderungen einer Modellbibliothek umgehen zu können. Zu diesen Änderungen zählen das Einfügen eines neuen Modells in eine Bibliothek, das Löschen eines existierenden Modells aus einer Bibliothek sowie die Änderung eines existierenden Modells.

Die Evaluation der LS3-Verfahren mit drei Datensätzen ergab eine sehr gute Qualität der Suchergebnisse im Sinne des F-Werts. Die F-Werte der LS3-Ansätzen lagen je nach Datensatz zwischen 0,86 und 0,93. Diese Werte sind auch im Vergleich mit fünf anderen Verfahren sehr gut, da diese bestenfalls F-Werte auf vergleichbarem Niveau erzielen, bei einem Datensatz jedoch deutlich schlechter abschneiden. Das zweitbeste Verfahren erzielt z. B. F-Werte im Bereich zwischen 0,78 und 0,95. Zudem wurde bei der Evaluation eine wesentlich größere Laufzeit der verglichenen Verfahren gegenüber den LS3-Ansätzen festgestellt. Während mit den LS3-Verfahren alle Ergebnisse in ca. 1,5 Sekunden berechnet werden konnten, benötigten die anderen Verfahren über eine Stunde, sodass die praktische Nutzbarkeit dieser Ansätze eingeschränkt ist. Darüber hinaus erreicht LS3-QueryAll auch für R-Precision und Precision-at-5 sehr gute Werte von 0,96 bis 0,99 und liegt damit ebenfalls vor den verglichenen Ansätzen.

Um mit den LS3-Verfahren gute Ergebnisse zu erzielen, ist es allerdings notwendig, dass in Modellen, die den gleichen Prozess beschreiben, zumindest teilweise übereinstimmende Wörter in den Beschriftungen enthalten sind. Bei der Analyse der Anfrageergebnisse der UA- und BR-Modelle fiel beispielsweise auf, dass diese Modelle verglichen mit den anderen Datensätzen wenige übereinstimmende Wörter enthalten.

Dadurch fielen die berechneten Ähnlichkeitswerte relativ gering aus, was zu schlechten Ergebnissen führte. Des Weiteren wurde aus der Evaluation ersichtlich, dass die Wahl der Dimensionalität des Vektorraums einen großen Einfluss auf die Qualität der Anfrageergebnisse hat. Bei einer zu geringen Dimensionalität ist die Qualität wesentlich niedriger als im besten Fall und die F-Werte sinken ebenfalls bei einer höheren Dimensionsanzahl. Bei einer passenden Dimensionalität zeigte sich hingegen die Güte der LS3-Verfahren im Vergleich mit Matching-basierten Ansätzen.

Kapitel 6

Anfragesprache für Prozessmodellbibliotheken

Nachdem sich Kapitel 4 mit dem Auffinden ähnlicher Aktivitäten in Prozessmodellen befasst hat und in Kapitel 5 ein Ansatz zur ähnlichkeitsbasierten Suche nach Prozessmodellen beschrieben wurde, wird in diesem Kapitel eine Möglichkeit erläutert, mit Hilfe einer Anfragesprache für Petri-Netz basierte Prozessmodelle nach Modellen zu suchen. Diese als *Petri Net Query Model Language* (PNQML) bezeichnete Sprache erlaubt gegenüber der ähnlichkeitsbasierten Suche eine größere Flexibilität bei der Formulierung von Anfragen. Es muss nicht notwendigerweise ein existierendes Modell für die Suche verwendet werden, sondern eine Suche kann beispielsweise auch nach Modellen erfolgen, die bestimmte Stellen oder Transitionen nicht enthalten. Neben der Wiederverwendung von Prozessmodellen können solche Suchen etwa auch in einer Funktion zur Autovervollständigung oder zur Ersetzung von Modellfragmenten eingesetzt werden (Markovic u. a. 2007).

Im restlichen Kapitel wird zunächst die zugrundeliegende Problemstellung formuliert (Kapitel 6.1). Anschließend wird in Abschnitt 6.2 auf verwandte Arbeiten eingegangen. Daraufhin wird die PNQML selbst in Kapitel 6.3 beschrieben. Dies beinhaltet eine Definition von Syntax, Semantik und Notation der PNQML sowie die Beschreibung eines Anfragemechanismus und einer Verknüpfung mit Prozesmodell-Matching Verfahren. Schließlich werden in Abschnitt 6.4 Unterschiede zu existierenden Verfahren diskutiert.

6.1 Problembeschreibung

Durch Prozessmodell-Matching und eine ähnlichkeitsbasierte Suche, wie sie in den Kapiteln 4 und 5 beschrieben wurden, werden noch nicht alle Möglichkeiten zur Suche nach Prozessmodellen abgedeckt. So ist es beispielsweise nicht möglich, explizit nach Modellen zu suchen, in denen eine bestimmte Transition nicht vorkommt, da keine Möglichkeit existiert, eine solche Restriktion auszudrücken. Zudem muss für eine ähnlichkeitsbasierte Suche ein existierendes Modell vorliegen, auf dessen Basis ein Suchvorgang durchgeführt wird. Ohne ein solches Modell ist eine Suche nicht möglich. Bei diesen Aspekten setzen *Anfragesprachen für Prozessmodellbibliotheken* an, die weitere Optionen für eine Suchanfrage bieten und das übergeordnete Thema „Suchen in Prozessmodellbibliotheken“ wiederum aus einer anderen Perspektive betrachten als Prozessmodell-Matching und ähnlichkeitsbasierte Suchverfahren.

In diesem Kapitel wird dazu eine Anfragesprache für Petri-Netz basierte Prozessmodelle entworfen, die *Petri Net Query Model Language* (PNQML). Dieser liegt die Idee des *Query-by-Example* von Zloof (1977) zugrunde. Es wird also keine rein textuelle Anfragesprache wie etwa SQL (*SQL92: Database Language SQL* 1992) im Datenbankbereich entwickelt, sondern es werden spezielle *Anfragemodelle* für eine Suche verwendet. Solche Anfragemodelle stellen Modelle dar, die eine Anfrage in der PNQML-Notation repräsentieren, die wiederum auf der Notation von Petri-Netzen basiert. Die Erstellung einer Anfrage soll durch die Beschreibung in einem Modell vereinfacht werden, da davon ausgegangen wird, dass Modellierer diese leicht erlernen können und im Vergleich zu einer textuellen Repräsentation durch die grafische Darstellung eine bessere Verständlichkeit einer Anfrage gewährleistet wird. Diese Anforderung nach einer intuitiven Spezifizierungsmöglichkeit von Anfragen wurde bereits verschiedentlich in der Literatur formuliert (siehe etwa (Markovic u. a. 2007) oder (Awad 2007)), allerdings noch nicht für Petri-Netz basierte Prozessmodelle umgesetzt.

Die Menge an Ergebnismodellen für ein Anfragemodell muss alle Bedingungen, die in diesem Anfragemodell beschrieben sind, erfüllen. Dies betrifft in Bezug auf die PNQML zwei der in Kapitel 2.2.2 erläuterten Dimensionen von Prozessmodellen: die Struktur eines Modells und die natürliche Sprache in Form von Beschriftungen. Zum einen sollen sich durch die PNQML Restriktionen bezüglich der Struktur eines Ergebnismodells formulieren lassen (z. B. welche Stellen und Transitionen durch Kanten verbunden sein sollen), zum anderen sollen durch die Verwendung natürlichsprachlicher Beschriftungen inhaltliche Restriktionen ausdrückbar sein. Daher muss

zur Beschreibung solcher Bedingungen Syntax, Semantik und Notation der PNQML definiert (Kapitel 6.3.1) sowie ein entsprechender Suchalgorithmus für eine Prozessmodellbibliothek entworfen werden (Kapitel 6.3.2).

Hinsichtlich der Beschriftungen stellt auch für diese Form der Suche die Wortwahl in einem Anfragemodell eine Herausforderung dar. Da jede Beschriftung in einem Anfragemodell eine Restriktion an Ergebnismodelle darstellt, kann eine „falsche“ Wortwahl dazu führen, dass relevante Modelle nicht in einem Anfrageergebnis enthalten sind. Bei dem intuitiv einfachen Vorgehen, Beschriftungen in Anfragemodellen auf identische Beschriftungen in potentiellen Ergebnismodellen zu überprüfen, führt z. B. schon die Verwendung des Synonyms „verschicken“ anstatt „versenden“ zu einem Ausschluss eines entsprechenden Modells. Die Anwendung einer String-Editier Distanz wie der Levenshtein-Distanz (Levenshtein 1966) führt dabei nur zu einer geringen Verbesserung (z. B. hohe Distanz zwischen „Lieferung versenden“ und „Paket versenden“). Daher wird für diese Problematik in Kapitel 6.3.3 ein Lösungsvorschlag beschrieben, der Prozessmodell-Matching Verfahren in die Beantwortung einer Anfrage miteinbezieht.

6.2 Verwandte Arbeiten

Verwandte Arbeiten in Bezug auf Anfragesprachen für Prozessmodelle lassen sich grundsätzlich in vier verschiedene Kategorien einteilen: (i) Anfragesprachen, die sich auf die Strukturdimension von Prozessmodellen beziehen; (ii) Anfragesprachen, die sich auf die Verhaltensdimension beziehen; (iii) Anfragesprachen, die auf textuellen Anfragen basieren; und (iv) Anfragesprachen, die Daten von Prozessinstanzen in Logs verwenden. Die Beschreibung der verwandten Arbeiten erfolgt zunächst unterteilt nach diesen Kategorien, während zum Abschluss dieses Abschnitts eine Übersicht der grundlegenden Arbeiten mit einem Vergleich der PNQML gegeben wird.

Auf weitere generische Möglichkeiten zur Suche nach Prozessmodellen wie beispielsweise durch Datenbankanfragesprachen wie SQL und den Einsatz generischer Anfragesprachen wie etwa (Delfmann u. a. 2015) wird dabei nicht eingegangen. Solche Ansätze verfügen typischerweise über keine grafische Darstellung für Prozessmodellanfragen bzw. es müssten solche Repräsentationen definiert werden, sodass sie im Folgenden nicht weiter berücksichtigt werden.

(i) Strukturbasierte Ansätze: Erste Arbeiten zu Anfragesprachen für Prozessmodelle wurden von Beeri u. a. (2006) publiziert (siehe auch (Beeri u. a. 2008) für eine erweiterte Beschreibung). Darin wird BP-QL als Sprache für BPEL-basierte[1] Modelle vorgestellt. Diese Sprache fokussiert auf einer an BPEL angelehnten, visuellen strukturellen Darstellung von Anfragen entsprechend der Query-by-Example Idee. Dabei sollen durch diese Sprache konkrete Fragen mit Hilfe von Prozessmodellen beantwortet werden können. Ein Beispiel einer solchen Frage ist, ob eine Bestellung ohne einen Kundenlogin möglich ist. Auf die Wiederverwendung von Modellen wird somit nicht explizit abgezielt.

Eine weitere frühe Arbeit von Markovic u. a. (2007) erläutert den Einsatz von logischen Ausdrücken zur Formulierung von Anfragen. Dabei liegt dem Anfragemechanismus eine Ontologie zugrunde, d. h., die Modelle müssen in einem entsprechenden Ontologieformat vorliegen. Die visuelle Spezifikation von Anfragen nach dem Query-by-Example Prinzip wird zwar erwähnt, jedoch nicht weiter ausgeführt. In (Markovic 2008) wird dieses Prinzip dadurch umgesetzt, dass Modellteile existierender Modelle selektiert und als Anfrage verwendet werden können.

Choi u. a. (2007) verwenden hingegen mit der IPM-PQL eine XML-basierte Anfragesprache. Diese erlaubt die Formulierung von Anfragen als XML-Dokumente, wobei sich die Anfragen auf die Struktur eines Prozessmodells und zugehörige Meta-Daten beziehen können. Zur Berechnung von Anfrageergebnissen werden aus den Anfrage-Dokumenten XQuery-Ausdrücke[2] erzeugt, durch die in einer XML-Datenbank gespeicherte Modelle gefunden werden können.

Awad (2007) beschreibt eine Sprache (BPMN-Q) für die Suche nach BPMN-Modellen in Bibliotheken. Dazu erlaubt die BPMN-Q eine visuelle Formulierung von Anfragen angelehnt an die Notation von BPMN. Zur Speicherung von Modellen und Beantwortung von Anfragen ist der Einsatz einer relationalen Datenbank angedacht. Diese Speicherform wird in (Awad und Sakr 2010) detaillierter ausgeführt. Dabei werden vier Relationen zur Speicherung von Modelldaten verwendet und ein Anfragemodell wird entsprechend zur Beantwortung in SQL transformiert. Diese Speicherart wurde darüber hinaus von Rharbi u. a. (2010) durch die Verwendung eines Objektorientierten Datenbanksystems angepasst. In (Awad u. a. 2008) wird der Anfrageme-

[1] http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html
[2] https://www.w3.org/TR/xquery-30

chanismus um eine Matching-Komponente erweitert, sodass die Beschriftungen von Anfragemodellen nicht exakt übereinstimmen müssen. Stattdessen werden auch ähnliche Aktivitäten, die über ein Matching-Verfahren bestimmt wurden, berücksichtigt. Eine weitere Ergänzung ist in (Sakr u. a. 2012) beschrieben. Es wird ein Vorgehen vorgeschlagen, um Anfragen, für die kein einzelnes Modell zutrifft, durch Modellteile aus verschiedenen Modellen zu beantworten. Eine Anfrage wird dabei in verschiedene Unteranfragen zerlegt, für die passende Modelle gesucht werden.

Francescomarino und Tonella (2008) spezifizieren mit der BPMN VQL eine Sprache, um nach Modellteilen in BPMN Modellen zu suchen. Dabei steht nicht die Suche nach Modellen in einer Bibliothek wie bei BPMN-Q im Vordergrund, sondern die Suche nach Modellteilen in einem Modell, die sich mit einem bestimmten Aspekt eines Prozesses befassen. Zur Auswertung einer Anfrage, die sich auch visuell darstellen lässt, wird auf ein eigenes Ontologieformat in Kombination mit SPARQL[3] zurückgegriffen. Des Weiteren wurde in (Müller 2009) eine Muster-basierte Sprache publiziert, um nach BPMN Modellen zu suchen. Aus solchen Mustern werden Regeln generiert, die zur Berechnung von Anfrageergebnissen verwendet werden. Zudem adaptieren Störrle und Acretoaie (2013) die ursprünglich für UML entworfene Sprache VQML für BPMN. Die VQML erlaubt eine visuelle Formulierung von Anfragen an BPMN Modelle, wobei allerdings auf eine konkrete Implementierung nicht eingegangen wird.

Yan u. a. (2012b) beschreiben darüber hinaus einen strukturbasierten Ansatz, der auf die grundlegende Graphstruktur von Prozessmodellen zurückgreift und eine entsprechende Anfragenotation als Graph spezifiziert. Die Graphstruktur wird dabei verwendet, um aus verschiedenen sogenannten „Features“ von Modellen, wie etwa Pfaden bestimmter Länge, eine Indexstruktur zu erstellen, die gegenüber verglichenen Anfrageverfahren eine effizientere Berechnung von Anfrageergebnissen erlaubt.

Die von Xiao u. a. (2009) beschriebene Sprache PNQL ist am ehesten mit der im folgenden Teilkapitel erläuterten PNQML vergleichbar. Die PNQL stellt ebenso eine visuelle Anfragesprache für Petri-Netze dar, wobei allerdings keine Möglichkeiten zur Spezifikation von nicht vorhandenen Knoten und Kanten besteht. Darüber hinaus wird keine Verknüpfung mit einem Prozessmodell-Matching Verfahren beschrieben, um Knoten mit ähnlichen Beschriftungen zu finden.

[3] https://www.w3.org/standards/semanticweb/query

(ii) Verhaltensbasierte Ansätze: Jin u. a. (2011) berücksichtigen die Verhaltensdimension von Prozessmodellen für den Entwurf der Anfragesprache BQL. Mit der BQL können Anfragen formuliert werden, die sich auf die Ausführungsreihenfolge von Aktivitäten beziehen. Beispielsweise können Modelle gesucht werden, in denen Aktivität A vor Aktivität B durchgeführt wird und in denen die Aktivitäten C und D nebenläufig ausgeführt werden können. Des Weiteren zielt auch der von Polyvyanyy u. a. (2014) entwickelte Ansatz auf Abfragen hinsichtlich der Verhaltensdimension ab. Mit diesem Ansatz sollen Modelle ermittelt werden können, die ein gemeinsames Auftreten von Aktivitäten in Prozessinstanzen ermöglichen. Das heißt, ein Modell ist dann im Ergebnis einer Anfrage enthalten, wenn dieses Modell eine Prozessinstanz zulässt, in der alle Aktivitäten einer Anfrage ausgeführt würden. Die Anfragesprache APQL erlaubt zudem weitere Ausdrucksmöglichkeiten bezüglich der Abfolge von Aktivitäten in Prozessinstanzen (ter Hofstede u. a. 2013). So kann beispielsweise abgefragt werden, ob bestimmte Aktivitäten in allen Instanzen nebenläufig ausgeführt werden. Darüber hinaus findet sich in (Polyvyanyy u. a. 2015) durch die PQL noch eine Erweiterung gegenüber der APQL mit einer an SQL angelehnten Syntax. Bei all diesen verhaltensbasierten Anfragesprachen besteht jedoch keine Möglichkeit einer visuellen Spezifikation einer Anfrage.

Eine weitere grafische Anfragesprache, die zu den verhaltensbasierten Sprachen gezählt werden kann, findet sich in (Oberweis und Sänger 1994). Diese Sprache erlaubt die Formulierung von Anfragen an sogenannte Simulationsdatenbanken, in denen Zustände von Simulationsdurchläufen auf Basis von Petri-Netzen gespeichert sind. Folglich könnten mit Hilfe dieser Sprache auch Simulationsdaten von Petri-Netz basierten Prozessmodellen abgefragt werden.

(iii) Textuelle Ansätze: Die Anfrageverfahren von (Shao u. a. 2009; Liu u. a. 2010) und (Wasser u. a. 2006) bieten Stichwort-basierte Suchmöglichkeiten. Wasser u. a. (2006) beschreiben ein Verfahren, um nach Informationen in Prozessmodellen zu suchen. Die gesuchten Informationen werden dabei über Stichwörter zu verschiedenen Kategorien wie dem Modellnamen oder dem Erstellungsdatum festgelegt. In (Shao u. a. 2009; Liu u. a. 2010) wird ein Verfahren vorgestellt, um nach hierarchischen Workflows zu suchen und darin zu Stichwörtern passende Tasks zu finden. Auch der Ansatz von Lincoln und Gal (2011) zielt auf die Beantwortung natürlichsprachlicher Anfragen ab. Hierbei wird ein Verfahren vorgestellt, welches in natürlicher Sprache formulierte „How to“ Anfragen durch Prozessmodelle in einer Bibliothek beantwortet. Jedoch

wird in keiner dieser Publikationen eine grafische Sprache zu Spezifikation von Anfragen beschrieben.

Goud u. a. (2006) beschreiben neben einer Stichwort-basierten Suche über Metadaten zu Modellen auch die Verwendung von Modelleigenschaften als Suchkriterien. So kann z. B. spezifiziert werden, dass nur Modelle gesucht sind, die einen Deadlock enthalten und beschränkt sind. Solche Eigenschaften werden bislang bei keinem anderen Suchverfahren berücksichtigt.

Darüber hinaus haben Leopold u. a. (2016) einen Ansatz entwickelt, um neben Modellinhalten auch damit verbundene ausführlichere textuelle Prozessbeschreibungen durchsuchen zu können. Dieser Ansatz basiert auf Technologien des Semantic Web, wobei dabei allerdings keine spezifische Anfragesprache für Prozessmodelle entworfen wird, sondern Anfragen mit Hilfe von SPARQL formuliert werden müssen. Somit ist eine Darstellung einer Anfrage in Form eines Anfragemodells nicht möglich.

(iv) Log-basierte Ansätze: Beheshti u. a. (2011) verfolgen einen anderen Ansatz als die zuvor erläuterten Sprachen. Durch ihre Erweiterung von SPARQL können Ereignisdaten von Prozessausführungen, die in einem Log vorliegen, abgefragt werden. Dies kann beispielsweise dazu genutzt werden, um alle Ereignisse zu einem Kunden zu ermitteln, die während Prozessausführungen aufgetreten sind. Hier steht demnach nicht die Wiederverwendung von Modellen im Fokus, sondern es wird auf die Analyse von Ereignislogs durch Anfragen abgezielt. In (de Murillas u. a. 2017) können mit Hilfe der entwickelten Sprache DAPOQ-Lang neben den Ausführungsdaten zu Aktivitäten auch zugehörige Datenobjekte in Anfragen berücksichtigt bzw. als Ergebnis ausgegeben werden.

Ein weiterer Ansatz, der auf Anfragen an Daten zu Prozessausführungen fokussiert, ist in (Momotko und Subieta 2004) beschrieben. Ziel ist eine flexiblere Definition von Bedingungen und Anforderungen in Prozessmodellen, um diese einfacher adaptieren zu können. Mit Hilfe der in (Momotko und Subieta 2004) erläuterten Anfragesprache BPQL können komplexe dynamische Bedingungen, die auf Daten zu Prozessausführungen referenzieren, in Prozessmodelle eingebunden werden, sodass diese während einer Prozessausführung überprüft werden können. Hier steht somit ebenfalls nicht die Wiederverwendung von Prozessmodellen im Vordergrund, sondern die Spezifizierung von Bedingungen in Prozessmodellen.

Tabelle 6.1 zeigt schließlich eine Übersicht der Charakteristika der existierenden Anfragesprachen für Prozessmodelle. In der Spalte Anfragesprache ist der Name der Sprache notiert. Die Spalten Visualisierung und Matching geben an, ob die Sprache eine grafische Formulierung von Anfragen in Form eines Modells erlaubt und ob ein Matching von Prozessmodellelementen bei der Berechnung von Anfrageergebnissen berücksichtigt wird. In der darauf folgenden Spalte ist festgehalten, ob die Anfragesprache für Prozessmodelle in unterschiedlichen Modellierungssprachen einsetzbar ist oder ob sie für eine bestimmte Sprache definiert wurde. Schließlich verdeutlichen die letzten beiden Spalten, auf welche Dimension eine Anfragesprache fokussiert und ob mit der Sprache auch nach fehlenden Elementen in einem Modell gesucht werden kann. Dazu zählen beispielsweise Möglichkeiten, um in einer Anfrage zu spezifizieren, dass zwischen zwei Aktivitäten keine Kante bzw. kein Pfad in einem Modell vorliegen soll oder dass ein Modell eine bestimmte Aktivität mit einer bestimmten Beschriftung nicht enthalten soll. Der Wert teilweise bedeutet dabei, dass solche Bedingungen für einige Bestandteile eines Modells bestimmt werden können, aber nicht für alle (z. B. Spezifikation nicht erlaubter Pfade, aber keine Möglichkeit nach Modellen zu suchen, die eine Aktivität nicht enthalten).

Hinsichtlich der PNQML sind die Anfragesprachen am nächsten verwandt, die ebenfalls die Struktur bei der Anfrageformulierung in den Mittelpunkt stellen (der oberste Abschnitt in Tabelle 6.1). Diese Sprachen zielen darauf ab, dass Modellierer, die bereits mit einer Modellierungssprache vertraut sind, Anfragen an Modellbibliotheken in Form eines Modells stellen können. Die anderen Sprachen zielen hingegen eher auf andere Zwecke ab wie z. B. eine allgemeine Modellsuche über Stichwörter oder die Analyse von Daten zu Prozessausführungen in Logs. Die Mehrheit der auf die Struktur fokussierenden Anfragesprachen bezieht sich auf BPMN, wohingegen für Petri-Netz basierte Prozessmodelle bislang nur die PNQL von (Xiao u. a. 2009) existiert.

Am engsten verwandt zur im folgenden Kapitel beschriebenen PNQML sind BPMN-Q von Awad (2007) und PNQL von Xiao u. a. (2009). Während sich die grundlegenden Ideen dieser beiden Sprachen auch in der PNQML wiederfinden, so unterscheidet sie sich doch im Detail. Im Vergleich zu BPMN-Q bezieht sich die PNQML nicht auf BPMN Modelle, sondern auf Petri-Netze und erlaubt darüber hinaus die Spezifikation von Aktivitäten, die nicht in einem Modell vorkommen sollen. Die grundlegende Matching Idee wird von BPMN-Q übernommen, allerdings ist es mit der PNQML möglich für einzelne Elemente eines Anfragemodells festzulegen, ob ein Matching stattfinden soll oder nicht. Bei BPMN-Q wird ein Matching grundsätzlich für alle Ele-

Referenz	Anfragesprache	Visualisierung	Matching	Sprachabhängig	Fokus	Fehlende Elemente
Beeri u. a. 2006; Beeri u. a. 2008	BP-QL	Ja	Nein	Nein	Struktur	Ja
Choi u. a. 2007	IPM-PQL basierend auf XML und XQuery	Nein	Nein	Nein	Struktur, Stichwörter	Nein
Awad 2007; Awad u. a. 2008; Awad und Sakr 2010	BPMN-Q	Ja	Ja	Ja	Struktur	Teilweise
Markovic u. a. 2007; Markovic 2008	Logischer Ausdruck und PI-Kalkulus über speziellem Ontologieformat von Modellen	Ja	Nein	Nein	Struktur	Nein
Francescomarino und Tonella 2008	BPMN VQL basierend auf SPARQL	Ja	Nein	Ja	Struktur	Ja
Xiao u. a. 2009	PNQL	Ja	Nein	Ja	Struktur	Nein
Müller 2009	PPML	Ja	Nein	Ja	Struktur	Teilweise
Yan u. a. 2012b	Graphnotation	Ja	Ja	Nein	Struktur	Teilweise
Störrle und Acretoaie 2013	VMQL	Ja	Nein	Ja	Struktur	Teilweise
Jin u. a. 2011	BQL	Nein	Ja	Ja	Verhalten	Ja
ter Hofstede u. a. 2013	APQL	Nein	Ja	Nein	Verhalten	Nein
Polyvyanyy u. a. 2014	Untanglings	Nein	Ja	Ja	Verhalten	Ja
Polyvyanyy u. a. 2015	PQL	Nein	Ja	Nein	Verhalten	Nein
Oberweis und Sänger 1994	Petri-Netz basiert	Ja	Nein	Ja	Verhalten	Ja
Wasser u. a. 2006	-	Nein	Nein	Nein	Stichwörter	Nein
Goud u. a. 2006	-	Nein	Nein	Ja	Stichwörter	Nein
Shao u. a. 2009; Liu u. a. 2010	-	Nein	Nein	Nein	Stichwörter	Nein
Lincoln und Gal 2011	-	Nein	Ja	Nein	Stichwörter, Verhalten	Nein
Leopold u. a. 2016	SPARQL-Anfragen über Modell und assoziierte Texte	Nein	Nein	Nein	Stichwörter	Nein
Momotko und Subieta 2004	BPQL	Nein	Nein	Abhängig von Meta-Modell	Logs	Nein
Beheshti u. a. 2011	FPSPARQL als Erweiterung zu SPARQL	Nein	Nein	Nein	Logs	Nein
de Murillas u. a. 2017	DAPOQ-Lang	Nein	Nein	Nein	Logs	Nein

Tab. 6.1: Übersicht von Anfragesprachen für Prozessmodelle

mente durchgeführt. Von der PNQL wird die Idee übernommen, Anfragen grafisch in Anlehnung an die Petri-Netz Notation darzustellen. Ergänzt wird die PNQL hingegen durch die Verknüpfung einer Anfrage mit Prozessmodell-Matching Verfahren, was in (Xiao u. a. 2009) nicht berücksichtigt ist.

6.3 Petri Net Query Model Language

In diesem Abschnitt wird eine Anfragesprache für Prozessmodelle, die durch Petri-Netze repräsentiert sind, beschrieben. Die *Petri Net Query Model Language* (PNQML) kann dazu verwendet werden, Anfragemodelle zu erstellen. Dazu wird zunächst in Kapitel 6.3.1 auf die Syntax, Semantik und Notation der einzelnen Bestandteile der PNQML eingegangen. Daraufhin wird in Kapitel 6.3.2 ein Suchalgorithmus vorgestellt, durch den basierend auf einer Prozessmodellbibliothek Ergebnismodelle zu einem Anfragemodell ermittelt werden können. Schließlich wird in Abschnitt 6.3.3 auf eine Ergänzung der PNQML durch Prozessmodell-Matching Ansätze eingegangen, um mit der Variabilität von Beschriftungen umgehen zu können.

6.3.1 Syntax, Semantik und Notation der PNQML

Für die Definition der Petri Net Query Model Language wird grundsätzlich auf die Ausführungen von Karagiannis und Kühn (2002) zu den Bestandteilen einer Modellierungssprache zurückgegriffen. Bei diesen Bestandteilen handelt es sich um Syntax, Semantik und Notation, was auch in Abbildung 6.1 dargestellt ist. Nach Karagiannis und Kühn (2002) beinhaltet die *Syntax* die Elemente und Regeln zur Erstellung von Modellen mit einer Sprache und definiert somit deren Grammatik. Die *Semantik*-Komponente einer Sprache beschreibt die Bedeutung einer Modellierungssprache sowie deren Elemente. Mit Hilfe einer *Notation* wird schließlich die visuelle Repräsentation einer Sprache definiert.

Hinsichtlich der Syntax der PNQML wird grundlegend von beschrifteten Workflow-Netzen nach Definition 2.7 ausgegangen, d. h., dass zur Modellierung eines Anfragemodells Stellen, Transitionen, deren Beschriftungen sowie Kanten zur Verfügung stehen. Darüber hinaus sind allerdings weitere Modellierungselemente wünschenswert, um eine größere Variabilität bei der Formulierung eines Anfragemodells zu erzielen. Dabei muss für Anfragemodelle die Workflow-Eigenschaft aus Definition 2.6

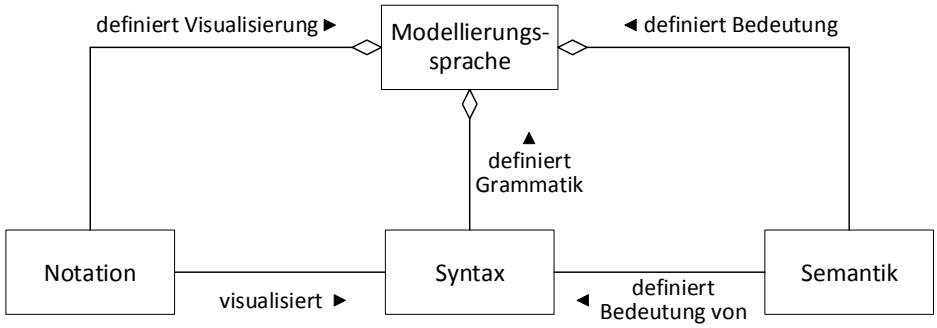


Abb. 6.1: Komponenten von Modellierungssprachen (in Anlehnung an Fill und Karagiannis (2013))

nicht zwingend erfüllt sein, da Anfragemodelle Bedingungen an potentielle Ergebnismodelle stellen und keine vollständigen Prozessmodelle darstellen müssen. Zu sich daraus ergebenden Implikationen für den Einsatz von Prozessmodell-Matching Verfahren siehe auch die Diskussion in Abschnitt 6.4.

Bezüglich der Kanten ergeben sich drei Ergänzungen, sodass nicht nur Kanten zwischen Stellen und Transitionen bzw. Transitionen und Stellen modelliert werden können. *Negative Kanten* stellen in gewissem Sinne eine Umkehrung von Kanten dar und sollen ausdrücken, dass Modelle gesucht sind, die eine solche Kante nicht enthalten. *Pfadkanten* sollen ausdrücken, dass zwischen zwei Knoten eines Modells ein Pfad existieren soll. Dadurch soll in einem Anfragemodell eine Verbindung zwischen zwei Knoten darstellbar sein, die auch als Folge mehrerer Kanten auftreten kann. Darüber hinaus beschreiben *Negative Pfadkanten* analog zu negativen Kanten eine Einschränkung und können dazu verwendet werden, nach Modellen zu suchen, die einen bestimmten Pfad nicht aufweisen.

Zusätzlich ergeben sich für Knoten und Beschriftungen folgende Ergänzungen. So stellen *negative Stellen* und *negative Transitionen* Stellen bzw. Transitionen dar, die nicht in den Ergebnismodellen zu einer Anfrage enthalten sein sollen. Ein *(negativer) variabler Knoten* drückt aus, dass in einem Anfragemodell ein Knoten mit einer bestimmten Beschriftung (nicht) existieren soll, unabhängig davon, ob es sich bei dem Knoten um eine Stelle oder eine Transition handelt. Zudem können Beschriftungen als *variable Beschriftungen* gekennzeichnet werden, sodass in einem Ergebnismodell zwar ein

entsprechender Knoten vorkommen muss, dessen Beschriftung allerdings einen beliebigen Inhalt enthalten kann.

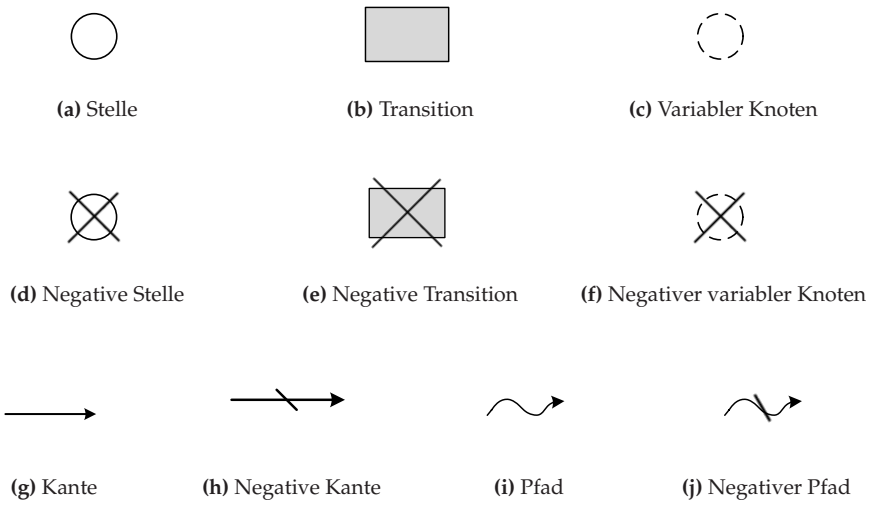
Zur formalen Definition der Syntax der Petri Net Query Model Language werden die in Definition 2.7 beschriebenen beschrifteten Workflow-Netze erweitert, sodass entsprechende Modelle als Anfragen verwendet werden können. Eine formale Definition dieser als *Petri-Netz Anfragemodell* bezeichneten Modelle findet sich in Definition 6.1. Zusätzlich sind im Folgenden eine informale textuelle Beschreibung der Semantik der Syntaxelemente sowie in Abbildung 6.2 eine visuelle Notation angegeben.

Definition 6.1: Petri-Netz Anfragemodell

Ein Petri-Netz Anfragemodell QM ist ein Tupel $QM = (N, A, \ell)$, wobei

- $N = \{S \cup T \cup NS \cup NT \cup VAR \cup NVAR\}$ eine endliche Menge von Knoten ist mit folgenden Knotentypen:
  - $S = \{s_1, s_2, \cdots, s_m\}$: Endliche Menge von Stellen,
  - $NS = \{ns_1, ns_2, \cdots, ns_n\}$: Endliche Menge von negativen Stellen,
  - $T = \{t_1, t_2, \cdots, t_o\}$: Endliche Menge von Transitionen,
  - $NT = \{nt_1, nt_2, \cdots, nt_p\}$: Endliche Menge von negativen Transitionen,
  - $VAR = \{var_1, var_2, \cdots, var_q\}$: Endliche Menge an variablen Knoten und
  - $NVAR = \{nvar_1, nvar_2, \cdots, nvar_r\}$: Endliche Menge an negativen variablen Knoten.
- $A = \{F \cup NF \cup P \cup NP\}$ eine endliche Menge von Verbindungen ist mit folgenden Verbindungstypen:
  - $F \subseteq (S \times T) \cup (T \times S)$: Menge von Kanten,
  - $NF \subseteq (S \times T) \cup (T \times S)$: Menge von negativen Kanten,
  - $P \subseteq (S \times S) \cup (S \times T) \cup (S \times VAR) \cup (T \times S) \cup (T \times T) \cup (T \times VAR)$: Menge von Pfaden und
  - $NP \subseteq (S \times S) \cup (S \times T) \cup (S \times VAR) \cup (T \times S) \cup (T \times T) \cup (T \times VAR)$: Menge von negativen Pfaden.

- $\ell : N \to \mathfrak{L}$ eine Beschriftungsfunktion ist, die jedem Knoten eine Beschriftung zuweist. $\mathfrak{L}$ enthält zusätzlich ein spezielles Element v, das eine variable Beschriftung ausdrückt.
- Darüber hinaus gilt:
  - Knoten aus NS, NT und $NVAR$ dürfen keine variable Beschriftung v enthalten.
  - Die Mengen S, T, NS, NT, VAR und $NVAR$ müssen paarweise disjunkt sein.
  - Die Mengen F, NF, P und NP müssen paarweise disjunkt sein.



(a) Stelle **(b)** Transition **(c)** Variabler Knoten

(d) Negative Stelle **(e)** Negative Transition **(f)** Negativer variabler Knoten

(g) Kante **(h)** Negative Kante **(i)** Pfad **(j)** Negativer Pfad

Abb. 6.2: Notation der Syntaxelemente für Petri-Netz Anfragemodelle nach Definition 6.1

Negative Transition / negative Stelle: Die als negative Transition bzw. negative Stelle bezeichneten Elemente eines Anfragemodells repräsentieren Stellen oder Transitionen mit einer Beschriftung, die nicht in einem Ergebnismodell enthalten sein dürfen. Ein Modellierer eines Anfragemodells kann dadurch explizit Ausschlusskriterien für Ergebnismodelle spezifizieren. Diese Stellen bzw. Transitionen werden dabei über die exakte Übereinstimmung von Beschriftungen identifiziert. In einem Anfragemodell vorhandene Stellen und Transitionen repräsentieren hingegen Knoten, die in einem

Ergebnismodell vorkommen müssen. Diese werden ebenso anhand der exakten Übereinstimmung der Beschriftungen erkannt.

Variabler Knoten / negativer variabler Knoten: Ein variabler Knoten kann dazu verwendet werden, nach einem Knoten mit einer spezifischen Beschriftung in einem Modell zu suchen, unabhängig davon, ob es sich dabei um eine Stelle oder eine Transition handelt. Nur Modelle, die einen solchen Knoten enthalten, werden in das Suchergebnis mit aufgenommen. Entsprechend kann durch einen negativen variablen Knoten – ohne den Knotentyp spezifizieren zu müssen – ein Knoten mit einer spezifischen Beschriftung beschrieben werden, durch den Modelle aus dem Suchergebnis ausgeschlossen werden, wenn sie einen Knoten mit einer übereinstimmenden Beschriftung enthalten.

Kante / negative Kante: Kanten zwischen zwei Knoten n_i und n_j in einem Anfragemodell bedeuten, dass diese Kante auch in den Ergebnismodellen vorhanden sein muss. Die Kanten werden dabei über die Beschriftungen der beteiligten Knoten identifiziert. Falls die Beschriftungen in dem Anfragemodell und einem Ergebnismodell exakt übereinstimmen und die Richtung der Kante gleich ist, wird diese Kante als übereinstimmend klassifiziert. Negative Kanten in einem Anfragemodell werden prinzipiell wie Kanten behandelt, nur dass diese Kanten in einem Ergebnismodell nicht vorkommen dürfen. Zusätzlich dürfen (negative) Kanten in einem Anfragemodell nur zwischen Stellen und Transitionen bzw. Transitionen und Stellen spezifiziert werden, um die Regeln von Petri-Netzen einzuhalten (siehe Definition 2.4).

Pfad / negativer Pfad: Kanten sind bei der Modellierung eines Anfragemodells insofern einschränkend, als dass nur aufeinander folgende Stellen und Transitionen miteinander verbunden werden können. Möchte ein Modellierer beispielsweise zwei Transitionen direkt miteinander über eine Kante verknüpfen, so ist dies nicht erlaubt. Dieser Fall ist aber sinnvoll, um ausdrücken zu können, dass zwei Knoten mit beliebigem Typ über einen Pfad miteinander verbunden sind, ohne genau spezifizieren zu müssen, wie lang dieser Pfad ist und welche Knoten Bestandteil dieses Pfads sein müssen.

Über das Pfad-Element eines Anfragemodells lassen sich solche Sachverhalte ausdrücken. Dieser Verbindungstyp kann verwendet werden, um mit Ausnahme negativer Stellen NS, negativer Transitionen NT und negativer variabler Knoten $NVAR$ beliebige Knoten miteinander zu verknüpfen. Eine Verbindung mit *negativen* Knotentypen wird ausgeschlossen, da eine solche Bedingung nie von einem Ergebnismodell erfüllt

werden kann. Negative Pfade können dazu eingesetzt werden, um auszudrücken, dass zwischen zwei Knoten kein Pfad existieren darf. Wie bei den zuvor beschriebenen Kanten werden die an einem Pfad beteiligten Knoten über ihre Beschriftungen identifiziert, d. h., die Beschriftungen der Knoten müssen exakt übereinstimmen.

Variable Beschriftung: Schließlich kann die spezielle Beschriftung v als eine Art Platzhalter für eine Beschriftung eines Knotens verwendet werden. Für einen Knoten mit variabler Beschriftung in einem Anfragemodell wird somit lediglich überprüft, ob in einem Ergebnismodell ein Knoten entsprechenden Typs existiert und ob mögliche weitere Restriktionen wie etwa Kantenverbindungen von diesem Knoten erfüllt werden. Daher können variable Beschriftungen z. B. dazu eingesetzt werden, nach strukturellen Mustern in Modellen zu suchen und dabei von den Beschriftungen zu abstrahieren. Sollte eine Beschriftung hingegen leer sein, wird in den gesuchten Modellen nur dann eine Übereinstimmung erkannt, wenn ebenfalls ein Knoten mit einer leeren Beschriftung existiert.

Zudem dürfen Knoten der Mengen NS, NT und $NVAR$ keine variable Beschriftung enthalten, da solche Bedingungen von jedem Modell einer Modellbibliothek erfüllt werden. Darüber hinaus muss ein Anfragemodell QM die Workflow-Eigenschaft aus Definition 2.6 nicht zwingend erfüllen, da in einem Anfragemodell auch Bedingungen durch voneinander unabhängige Modellteile definiert werden können sollen und diese Eigenschaft für die Berechnung von Ergebnismodellen nicht benötigt wird. Weitere Modellierungselemente von Petri-Netzen wie beispielsweise Marken oder Gewichte werden aktuell in der PNQML nicht berücksichtigt, könnten bei Bedarf aber zu Definition 6.1 hinzugefügt werden.

6.3.2 Suchalgorithmus

Algorithmus 6.1 zeigt den Ablauf zur Berechnung der Ergebnismodelle für ein Anfragemodell. Dieser Algorithmus benötigt zur Berechnung von Ergebnismodellen RM ein Anfragemodell QM nach Definition 6.1 sowie eine Modellbibliothek **M**. Im Wesentlichen wird durch den Algorithmus für alle Modelle in **M** überprüft, ob die in dem Anfragemodell spezifizierten Knoten und Verbindungen enthalten sind.

In den Zeilen 1–7 werden zunächst einige Variablen definiert, die für eine effiziente Berechnung der Anfrageergebnisse notwendig sind. Die Variable *continue* gibt dabei

Algorithmus 6.1 : PNQML Suchalgorithmus

```
   input  : Query model QM, model collection M.
   output : Result models RM.
1  boolean continue = true;
2  boolean include = true;
3  List nonVariableNodes = getNonVariableNodes (QM);
4  List variableNodes = getVariableNodes (QM);
5  List edges = getEdges (QM);
6  List paths = getPaths (QM);
7  Set RM = ∅;
   /* Calculation of query result */
8  for each (model m in M) do
9      if (continue == true) then
10         for each (node n in nonVariableNodes) do
11             if (!m.findNonVar(n)) then
12                 continue = false; include = false; break;
13             end
14         end
15     end
16     if (continue == true) then
17         for each (node n in variableNodes) do
18             if (!m.findVar(n)) then
19                 continue = false; include = false; break;
20             end
21         end
22     end
23     if (continue == true) then
24         for each (edge e in edges) do
25             if (!m.find(e)) then
26                 continue = false; include = false; break;
27             end
28         end
29     end
30     if (continue == true) then
31         for each (path p in paths) do
32             if (!m.find(p)) then
33                 continue = false; include = false; break;
34             end
35         end
36     end
37     if (include == true) then
38         Add model m to RM;
39     end
40 end
41 return RM;
```

an, ob eine weitere Überprüfung eines Modells m aus **M** notwendig ist. Sollte *continue* während des Ablaufs den Wert *false* erhalten, so kann eine weitere Überprüfung abgebrochen werden, da in einem vorigen Schritt bereits eine der Bedingungen aus

QM nicht durch das Modell m erfüllt werden konnte. Über die Variable *include* wird zudem festgelegt, ob ein Modell m in die Menge der Ergebnismodelle RM aufgenommen werden soll.

Die Variable *nonVariableNodes* enthält alle Modellelemente aus QM, die zu den Mengen S, NS, T und NT sowie VAR und $NVAR$ aus Definition 6.1 zählen und keine variable Beschriftung besitzen. In der Variable *variableNodes* sind hingegen alle Modellelemente aus S, T und VAR enthalten, für die eine variable Beschriftung definiert wurde. Schließlich sind in den Variablen *edges* und *paths* die Kanten bzw. Pfade aus QM gespeichert. Diese Aufteilung der einzelnen Modellelemente auf die verschiedenen Variablen erfolgt, um bei der anschließenden Überprüfung, ob ein Modell m alle Bedingungen aus QM erfüllt, die effizient zu bestimmenden Restriktionen möglichst frühzeitig zu überprüfen. Beispielsweise muss für die Bestimmung, ob ein Modell m eine Transition mit einer bestimmten Beschriftung enthält, lediglich eine Liste aller Transitionen von m untersucht werden. Um hingegen zu bestimmen, ob in einem Modell ein bestimmter Pfad zwischen zwei Knoten existiert, ist etwa eine Erreichbarkeitsanalyse mit Hilfe des Dijkstra Algorithmus notwendig.[4]

Die Berechnung der Ergebnismodelle findet nach der Definition der Variablen in den Zeilen 8–40 statt. Dazu wird für jedes Modell der Bibliothek separat evaluiert, ob es alle Bedingungen des Anfragemodells erfüllt. Zunächst werden in den Zeilen 10–14 die nicht-variablen Knoten untersucht. Für diese Knoten wird in der Funktion *findNonVar(n)* bestimmt, ob die entsprechende Restriktion durch das Modell m erfüllt wird. D. h., dass für Knoten aus S und T Stellen und Transitionen mit einer identischen Beschriftung in m existieren bzw. dass für Knoten aus NS und NT keine Stellen bzw. Transitionen mit identischer Beschriftung in m vorkommen.[5] Für die verbleibenden Knoten aus VAR und $NVAR$ wird daraufhin überprüft, ob Knoten mit einer identischen Beschriftung in m existieren bzw. nicht existieren. Sollte eine der Bedingungen nicht erfüllt sein, werden die Variablen *continue* und *include* auf *false* gesetzt und die Schleife wird verlassen. Weitere Überprüfungen sind nicht notwendig, da m nicht alle Bedingungen von QM erfüllen kann und somit nicht in RM enthalten ist.

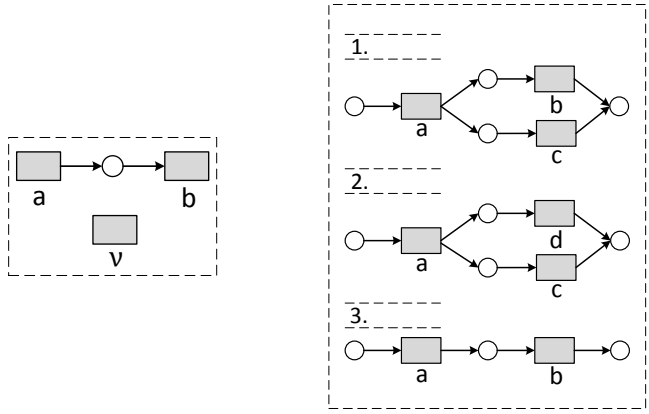
[4] Für diese Ausführungen wird angenommen, dass keine zusätzlichen Indexstrukturen oder Ähnliches verwendet werden, um die Berechnungen zu verkürzen.

[5] Sollten zwei oder mehr Knoten aus QM die gleiche Beschriftung aufweisen, so müssen auch in den Ergebnismodellen entsprechend viele Knoten mit diesen Beschriftungen vorhanden sein.

Anschließend werden in den Zeilen 17–21 die variablen Knoten überprüft. In der Funktion *findVar(n)* wird für diese Knoten festgestellt, ob in dem Modell m mindestens so viele weitere Knoten des entsprechenden Typs vorhanden sind, die noch nicht im vorigen Schritt auf Knoten mit fester Beschriftung abgebildet wurden. D. h. es müssen mindestens so viele weitere Stellen und Transitionen in m existieren, sodass jede Stelle und Transition aus QM mit variabler Beschriftung auf eine von ihnen abgebildet werden kann. Darüber hinaus müssen schließlich noch mindestens so viele weitere Knoten in m vorhanden sein, sodass jeder Knoten aus VAR mit variabler Beschriftung auf einen dieser Knoten abgebildet werden kann. Sollten nicht genügend passende Knoten in m vorhanden sein, werden *continue* und *include* auf *false* gesetzt und die Überprüfung von m wird abgebrochen, da nicht alle Bedingungen aus QM erfüllt werden können. In diesem Fall kann mindestens einer der Knoten aus *variableNodes* nicht auf einen passenden Knoten in m abgebildet werden.

Daraufhin wird in den Zeilen 24–28 überprüft, ob alle in QM vorhandenen Kanten zwischen zwei Knoten auch in m existieren bzw. ob negative Kanten aus QM nicht in m vorhanden sind. Dies wird schließlich auch für die in QM definierten (negativen) Pfade in den Zeilen 31–35 durchgeführt. Für (negative) Kanten zwischen zwei Knoten ohne variable Beschriftungen kann dies mit Hilfe der bereits zuvor bestimmten Abbildungen der Knoten von QM auf Knoten von m bestimmt werden. Sollten entsprechende Kanten aus QM in m vorhanden bzw. nicht vorhanden sein, erfüllt m diese Bedingungen. Für Pfade zwischen zwei Knoten ohne variable Beschriftungen muss in m überprüft werden, ob der Endknoten des Pfads von dem Startknoten aus erreichbar oder im Falle eines negativen Pfads nicht erreichbar ist.

Sollten zusätzlich in QM (negative) Kanten und (negative) Pfade definiert sein, die Knoten mit variablen Beschriftungen beinhalten, muss eine Abbildung aller Knoten mit variablen Beschriftungen aus QM auf entsprechende Knoten in m gefunden werden. Dabei ist zu berücksichtigen, dass Stellen aus QM nur auf Stellen in m abgebildet werden. Analog gilt auch für Transitionen in QM, dass diese nur auf Transitionen in m abgebildet werden dürfen. Sollte keine Abbildung gefunden werden können, durch die alle Bedingungen durch (negative) Kanten und (negative) Pfade mit variablen Beschriftungen erfüllt werden, wird die Überprüfung abgebrochen und m nicht zur Menge der Ergebnismodelle RM hinzugefügt. Ist der Wert der Variable *include* hingegen nach allen Überprüfungen immer noch *true*, bedeutet das, dass das Modell m alle Bedingungen aus QM erfüllt. Daher wird m zu der Menge an Ergebnismodellen RM hinzugefügt.



(a) Anfragemodell **(b)** Modellbibliothek

Abb. 6.3: Beispiel für ein PNQML-Anfragemodell

In Abbildung 6.3a ist ein Beispiel für ein PNQML-Modell dargestellt. Dadurch werden Modelle gesucht, die zwei Transitionen mit den Beschriftungen „a“ und „b“ sowie eine weitere Transition mit einer variablen Beschriftung enthalten. Zudem müssen die Transitionen a und b durch eine unbeschriftete Stelle und zwei Kanten miteinander verbunden sein. Aus der in Abbildung 6.3b dargestellten Modellbibliothek erfüllt lediglich das 1. Modell alle diese Restriktionen. In diesem Modell sind ebenfalls zwei Transitionen a und b enthalten, die durch eine unbeschriftete Stelle und zwei Kanten verbunden sind. Zusätzlich existiert eine Transition c, auf die die Transition mit der variablen Beschriftung aus dem Anfragemodell abgebildet werden kann. Das 2. Modell ist dagegen nicht in dem Anfrageergebnis enthalten, da es keine Transition b sondern eine Transition mit der Beschriftung „d“ enthält. Das 3. Modell erfüllt ebenfalls nicht alle Bedingungen aus dem Anfragemodell, da eine dritte Transition fehlt.

6.3.3 Erweiterung um Prozessmodell-Matching

Bei Anfragen mit der PNQML existiert grundsätzlich die Problematik, dass nur Modelle in einem Anfrageergebnis enthalten sind, in denen die Beschriftungen identisch sind zu denen des PNQML-Modells. Dies ist insofern einschränkend als dass bereits die Verwendung von synonymen Wörtern in Modellen dazu führt, dass diese nicht in ein Anfrageergebnis aufgenommen werden. Genereller ausgedrückt bedeutet dies,

dass die Berücksichtigung von *ähnlichen* Knoten bei der Berechnung von Anfrageergebnissen sinnvoll sein kann. In diesem Abschnitt wird daher ein Ansatz beschrieben, der Prozessmodell-Matching verwendet, um diese Problematik zu verringern.

Bei der Integration von Prozessmodell-Matching in die PNQML sind verschiedene Aspekte zu klären, die im Folgenden erläutert werden. Zu diesen Aspekten zählen folgende Fragestellungen: (i) Ist der Einsatz existierender Matching-Verfahren möglich und welche Änderungen müssen vorgenommen werden? (ii) Für welche Elemente der PNQML ist es sinnvoll, ein Matching durchführen zu können? (iii) Wie kann ein Matching in die Ergebnisberechnung für PNQML-Modelle integriert werden?

Einsatz von Prozessmodell-Matching: Wie in Kapitel 4 erläutert wurde, können durch Prozessmodell-Matching Verfahren ähnliche Aktivitäten in Prozessmodellen bestimmt werden. Diese Verfahren können daher prinzipiell in Anfragesprachen wie der PNQML verwendet werden, um die Einschränkung auf exakt übereinstimmende Beschriftungen zu vermeiden. Dabei ist allerdings zu berücksichtigen, dass ein Matching typischerweise nur für die Aktivitäten der Modelle durchgeführt wird. Somit müssen die Verfahren erweitert werden, sodass auch andere Modellelemente wie z. B. Stellen oder negative Transitionen berücksichtigt werden können.

Ein weiterer Aspekt, den es zu überdenken gilt, betrifft die Berücksichtigung der Verhaltens- und Strukturdimensionen von Prozessmodellen (siehe Kapitel 2.2.2). Für diese Dimensionen müssen entweder die aktuell verwendeten Ähnlichkeitsberechnungen angepasst oder nicht mehr eingesetzt werden. Warum dies notwendig ist, wird anhand von Abbildung 6.4 und dem Triple-S Algorithmus zum Prozessmodell-Matching (Kapitel 4.3.1) erläutert. Die Berechnungen zur Dimension natürliche Sprache in Form der Beschriftungen können demgegenüber übernommen werden, da durch sie die Matching-Verfahren erkennen, ob zwei Aktivitäten die gleichen Tätigkeiten in der realen Welt beschreiben.

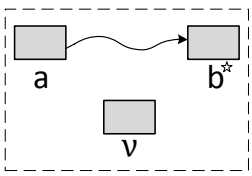


Abb. 6.4: Eine zweite Beispielanfrage mit der PNQML

In Abbildung 6.4 ist das Anfragemodell aus Abbildung 6.3a geringfügig geändert worden. Statt die beiden Transitionen a und b durch zwei Kanten und eine Stelle miteinander zu verbinden, sind sie nun durch einen Pfad miteinander verbunden. Des Weiteren wird durch den Stern kenntlich gemacht, dass für die Transition b bei der Berechnung von Ergebnismodellen auch ähnliche Transitionen berücksichtigt werden sollen. Bei den folgenden Ausführungen wird zudem davon ausgegangen, dass die Transition b aus dem zweiten Anfragemodell und die Transition d aus dem zweiten Modell der Modellbibliothek (siehe Abbildung 6.3b) die gleiche Aktivität beschreiben und sich lediglich die Beschriftungen unterscheiden.

Wenn davon ausgegangen wird, dass nur die Beschriftungen verglichen werden und dass die Ähnlichkeit der Beschriftungen „b“ und „d“ nach den Berechnungen des Triple-S Algorithmus ausreichend hoch ist, wäre auch das zweite Modell der Modellbibliothek in dem Anfrageergebnis enthalten. Soll hingegen auch die strukturelle Ähnlichkeitsberechnung des Triple-S Algorithmus berücksichtigt werden, ist unklar wie hoch die Ähnlichkeit bezüglich dieser Dimension ist, da nicht definiert ist, wie Pfade aus einer PNQML-Anfrage berücksichtigt werden können. In Bezug auf den Triple-S Algorithmus ist insbesondere nicht festgelegt, wie die Anzahl an Kanten eines Pfads in einem PNQML-Modell bestimmt wird, um die relative Positionsberechnung durchführen zu können.

In diesem Zusammenhang tritt darüber hinaus die Problematik auf, dass PNQML-Anfragen keine zusammenhängenden Graphen mit Start- und Endknoten bilden müssen. Eine entsprechende Restriktion würde die Flexibilität bei der Formulierung von Anfragen reduzieren. Diese Flexibilität führt allerdings dazu, dass die relative Position von Knoten nach der Triple-S Definition nicht bestimmt werden kann.

Hinsichtlich der Verhaltensdimension tritt eine ähnliche Problematik auf, da etwa für eine Stelle, die jeweils durch einen Pfad mit zwei Transitionen verbundenen ist, keine Ausführungsreihenfolge für die Transitionen definiert ist. Es wird lediglich auf struktureller Ebene festgelegt, dass die Stelle durch eine Folge von Kanten mit den Transitionen verbunden sein muss. Daher können sie theoretisch nacheinander, exklusiv zueinander oder nebenläufig durchgeführt werden. Für Matching-Verfahren, die die Ähnlichkeit von Aktivitäten basierend auf der Ausführungsreihenfolge bestimmen, ist somit nicht festgelegt, wie eine Berechnung durchzuführen ist.

Aufgrund dieser Schwierigkeiten lassen sich Prozessmodell-Matching Ansätze nur einsetzen, wenn auf die Berücksichtigung der Struktur- und Verhaltensdimensionen

verzichtet wird. Eine Alternative dazu ist die Anpassung von Matching-Verfahren an die Besonderheiten, die sich durch den Einsatz in einer Anfragesprache ergeben, sodass diese Besonderheiten entsprechend bei der Berechnung von Ähnlichkeitswerten berücksichtigt werden. Für die beiden in Kapitel 4 beschriebenen Triple-S Algorithmen hat dies zur Folge, dass die strukturellen Berechnungen entfernt werden müssen, um die Algorithmen in der PNQML verwenden zu können.

PNQML Matching-Elemente: Prinzipiell sind alle Knotentypen aus Definition 6.1 dazu geeignet, um bei der Ergebnisberechnung entsprechende Matches in den Modellen einer Modellbibliothek bestimmen zu können. Allerdings ist dies bei Knoten mit variabler oder leerer Beschriftung nicht möglich, da sich aufgrund fehlender Beschriftungen keine Matches berechnen lassen. Für negative Knoten, die ausdrücken, dass ein bestimmter Knoten nicht in einem Ergebnismodell vorhanden sein soll, sollte jedoch die Möglichkeit bestehen, Matches zu finden. Dadurch wird auch bei diesen Knotentypen die Variabilität der Formulierung in Beschriftungen erhöht.

Zusätzlich müssen die Triple-S Ansätze, die nur für Transitionen entsprechende Matches finden, so erweitert werden, dass auch zu Stellen und variablen Knoten entsprechende, ähnliche Knoten bestimmt werden können. Dazu muss Definition 4.2 so ergänzt werden, dass die Menge an Matches Λ nicht nur Tupel von Transitionen (T_i, T_j) enthält, sondern dass darin auch Tupel von Stellen (S_i, S_j) und Tupel von variablen Knoten (VAR_i, S_j) oder (VAR_i, T_j) enthalten sind. Zudem sollten auch für negative Knoten aus einem Anfragemodell Matches berechnet werden können, weshalb zusätzlich folgende Arten von Tupeln in Λ enthalten sein dürfen: (NT_i, T_j), (NS_i, S_j), $(NVAR_i, S_j)$ und $(NVAR_i, T_j)$.

Integration in die Ergebnisberechnung: Zur Darstellung, ob für einen Knoten Matches bei der Ergebnisberechnung berücksichtigt werden sollen, wird ein Stern verwendet (siehe Abbildung 6.4). Die Festlegung von Schwellenwerten zur Bestimmung von Matches wird für jeden Knoten einzeln zugelassen, um eine möglichst große Freiheit bei der Anfragenspezifikation zu erlauben. Zusätzlich kann allerdings für eine einfachere Festlegung von Schwellenwerten ein Standardwert vergeben werden, der immer dann genutzt wird, wenn kein anderer Wert angegeben wird. Die Definition eines Anfragemodells mit Matching ergibt sich schließlich folgendermaßen:

Definition 6.2: Petri-Netz Anfragemodell mit Matching

Ein Anfragemodell mit Matching QMM ist ein Tupel $QMM = (QM, m, t)$. Bei QM handelt es sich um ein Anfragemodell nach Definition 6.1. Für die Knoten $Z \subseteq N$, die keine variable oder leere Beschriftung enthalten, wird durch die Funktion m bestimmt, ob ein Matching durchgeführt werden soll oder nicht: $m : Z \rightarrow \{1, 0\}$. Die Funktion t weist jedem Knoten aus $Y := \{z \in Z \mid m(z) = 1\}$, für den durch die Funktion m ein Wert von 1 festgelegt wurde, einen individuellen Schwellenwert für die Match-Berechnung zu: $t : Y \rightarrow [0, 1]$.

Hinsichtlich des Algorithmus zur Berechnung von Anfrageergebnissen ergeben sich durch die Integration von Matches ebenso Änderungen. Der angepasste Ablauf ist in Algorithmus 6.2 dargestellt. Zunächst werden im Gegensatz zu dem ursprünglichen Algorithmus in den Variablen *nonVariableNodes* und *variableNodes* nur die Knoten gespeichert, die nicht zum Matching markiert sind. In der Variablen *matchNodes* sind hingegen die Knoten enthalten, für die Matches ermittelt werden sollen. Die weiteren Variablen sind wie zuvor in Algorithmus 6.1 definiert.

Eine weitere Änderung betrifft die Berechnung von Matches für die entsprechenden Knoten in Zeile 21. In der Methode *findMatches()* werden zu einem Knoten n Matches in dem potentiellen Ergebnismodell m ermittelt, wozu beispielsweise eines der Triple-S Verfahren aus Kapitel 4 verwendet werden kann. Diese Matches müssen zu dem Knoten n jeweils einen Ähnlichkeitswert aufweisen, der gleich groß oder über dem für diesen Knoten festgelegten Schwellenwert liegt. Dabei ist insbesondere zu berücksichtigen, dass ein Knoten n aus dem Anfragemodell auf mehrere Knoten eines Ergebnismodells abgebildet werden kann. Dies muss bei der Überprüfung der Kanten- und Pfadbedingungen berücksichtigt werden. Sollte zu einem der Knoten aus $matchNodes$ kein passender Knoten gefunden werden können, so ist diese Bedingung des Anfragemodells nicht erfüllt. Daher werden die weiteren Überprüfungen für das zugehörige Modell m nicht mehr durchgeführt und m wird nicht der Menge an Ergebnismodellen hinzugefügt.

In den *find*-Methoden in den Zeilen 26 und 31 wird wie in Algorithmus 6.1 für die Kanten und Pfade des Anfragemodells überprüft, ob entsprechende Kanten bzw. entsprechende Pfade in dem Modell m vorhanden ist. Eine Besonderheit ergibt sich dabei, wenn für einen Knoten aus $matchNodes$ mehr als ein entsprechender Knoten

Algorithmus 6.2 : PNQML Suchalgorithmus mit Matching

```
   input  : Query model QMM, model collection M.
   output : Result models RM.
1  boolean continue = true boolean include = true;
2  List nonVariableNodes = getNonVariableNodes (QMM);
3  List variableNodes = getVariableNodes (QMM);
4  List matchNodes = getMatchNodes (QMM);
5  List edges = getEdges (QMM);
6  List paths = getPaths (QMM);
7  Set RM = ∅;
   /* Calculate query result */
8  for each (model m in M) do
9      if (continue == true) then
10         for each (node n in nonVariableNodes) do
11             if (!m.findNonVar(n)) then continue = false; include = false; break;
12         end
13     end
14     if (continue == true) then
15         for each (node n in variableNodes) do
16             if (!m.findVar(n)) then continue = false; include = false; break;
17         end
18     end
19     if (continue == true) then
20         for each (node n in matchNodes) do
21             if (!m.findMatches(n)) then continue = false; include = false; break;
22         end
23     end
24     if (continue == true) then
25         for each (edge e in edges) do
26             if (!m.find(e)) then continue = false; include = false; break;
27         end
28     end
29     if (continue == true) then
30         for each (path p in paths) do
31             if (!m.find(p)) then continue = false; include = false; break;
32         end
33     end
34     if (include == true) then
35         Add model m to RM;
36     end
37     return RM;
38 end
```

in m ermittelt wurde. In einem solchen Fall müssen die Verbindungen aus dem Anfragemodell jeweils nur für einen entsprechenden Knoten und nicht für alle Knoten vorliegen. Dies ist notwendig, da bei einer Abbildung von einem Knoten auf mehrere entsprechende Knoten die zugehörige Graphstruktur nicht eindeutig definiert ist. So können die Knoten etwa auf einem Pfad oder auf unterschiedlichen Pfaden liegen.

Für Knoten aus $matchNodes$, die Bestandteil von negativen Kanten bzw. negativen Pfaden sind, gilt hingegen, dass jeder darauf abgebildete Knoten in m nicht über eine entsprechende Kante oder einen entsprechenden Pfad verbunden sein darf.

Sollten z. B. zu dem Knoten b aus dem Anfragemodell in Abbildung 6.4 die beiden Knoten c und d aus Abbildung 6.5 Matches sein, muss für die Kante aus dem Anfragemodell überprüft werden, ob eine entsprechende Verbindung in dem Ergebnismodell existiert. Dies ist für das Modell in Abbildung 6.5 der Fall, da ein Pfad von dem Knoten a zu dem Knoten c vorhanden ist. Obwohl also der Knoten d nicht über einen Pfad von a aus erreicht werden kann, erfüllt dieses Modell die Kantenbedingung.

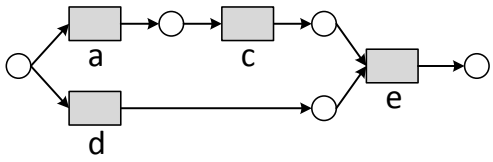


Abb. 6.5: Ein mögliches Ergebnismodell zum Anfragemodell aus Abbildung 6.4

Zusammenfassend ergibt sich, dass für Kanten und Pfade in Anfragemodellen, bei denen für einen oder beide Knoten Matches berücksichtigt werden sollen, lediglich eine Kante bzw. ein Pfad zwischen den abgebildeten Knoten vorhanden sein muss. Bei negativen Kanten oder Pfaden darf hingegen zwischen keinen abgebildeten Knoten eine entsprechende Verbindung existieren. Sollte eine der Kanten- oder Pfadbedingungen nicht erfüllt werden, wird die Überprüfung abgebrochen und das Modell m nicht zur Menge der Ergebnismodelle hinzugefügt. Falls alle Bedingungen erfüllt sein sollten, wird m hingegen als Ergebnismodell in die Menge RM eingefügt.

6.4 Diskussion

Im Folgenden wird der im vorigen Teilkapitel beschriebene Ansatz zur Kombination von Prozessmodell-Matching und PNQML mit den beiden anderen in Tabelle 6.1 aufgeführten Anfragesprachen, die ebenfalls eine visuelle Darstellung eines Anfragemodells mit Matching verbinden, verglichen. Bei dem von Awad u. a. (2008) vorgestellten Ansatz handelt es sich um eine Erweiterung der Anfragesprache BPMN-Q und Yan

u. a. (2012b) verwenden eine allgemeine Graphdarstellung von Prozessmodellen. Zudem wird auf die Ausdrucksmächtigkeit der PNQML eingegangen.

Ein genereller Unterschied zu (Awad u. a. 2008) besteht darin, dass die PNQML eine Spezifizierung von Knoten erlaubt, für die Matches bei der Ergebnisberechnung berücksichtigt werden sollen. In (Awad u. a. 2008) werden hingegen jeweils für alle Aktivitäten einer Anfrage ähnliche Aktivitäten aus einer Prozessmodellbibliothek bestimmt. Zur Ermittlung ähnlicher Aktivitäten wird dabei auf einen rein textuellen Vergleich von Aktivitätsbeschriftungen zurückgegriffen. In (Awad u. a. 2008) wird damit ebenfalls auf einen strukturellen Vergleich zur Berechnung von Matches verzichtet. Auch wenn es nicht explizit erwähnt wird, so ist zu vermuten, dass dafür ebenso die in Abschnitt 6.3.3 genannten Gründe ausschlaggebend waren. Zusätzlich wird die Festlegung von Schwellenwerten für ähnliche Aktivitäten unterschiedlich gehandhabt. Während bei der PNQML für jede Aktivität ein individueller Schwellenwert festgelegt werden kann, wird in (Awad u. a. 2008) nur ein Schwellenwert für alle Aktivitäten genutzt. Ein weiterer Unterschied betrifft die Berechnung von Ergebnissen. Während das Verfahren aus (Awad u. a. 2008) aus einer Anfrage weitere Anfragen generiert, werden in dem PNQML-Algorithmus 6.2 keine neuen Anfragen erzeugt. Stattdessen werden die Matches bei jedem Vergleich eines Prozessmodells mit einem Anfragemodell berücksichtigt. Dadurch ergeben sich in (Awad u. a. 2008) so viele Anfragen, wie es unterschiedliche Kombinationen von ähnlichen Aktivitäten gibt. Dies führt potentiell zu einer langen Laufzeit, um Anfrageergebnisse zu berechnen. Darüber hinaus werden in den Ausführungen in (Awad u. a. 2008) keine komplexen Matches berücksichtigt, was unrealistisch erscheint. Diese werden hingegen bei der Berechnung von Ergebnissen mit der PNQML berücksichtigt.

Ein grundlegender Unterschied zwischen der PNQML und der Anfragedarstellung in (Yan u. a. 2012b) betrifft die Modellierung von Anfragen. Während die PNQML auf Petri-Netzen basiert, wird in (Yan u. a. 2012b) eine allgemeine Graphrepräsentation für Anfragen verwendet. Dadurch kann dieser Ansatz für Modelle in unterschiedlichen Modellierungssprachen verwendet werden, allerdings wird ein Mechanismus benötigt, der für eine Anfrage überprüft, ob ein Modell in einer bestimmten Modellierungssprache in das Suchergebnis aufgenommen werden soll.

Hinsichtlich des Matchings von Knoten wird in (Yan u. a. 2012b) ein vergleichsweise einfacher Ansatz eingesetzt, der lediglich überprüft, ob die Wörter eines Knotens in

einem Anfragemodell mit den Wörtern eines Knotens in einem potentiellen Ergebnismodell übereinstimmen. Der Einsatz eines Prozessmodell-Matching Verfahrens wird ansonsten nicht weiter betrachtet. Darüber hinaus werden ebenso wie bei (Awad u. a. 2008) keine komplexen Matches berücksichtigt, sodass die PNQML die einzige Anfragesprache für Prozessmodelle ist, für die diese Aspekte diskutiert wurden.

Bezüglich der Ergebnisberechnung stimmen hingegen beide Anfragesprachen hinsichtlich der grundlegenden Idee überein. So werden wie bei der PNQML auch in (Yan u. a. 2012b) Modellteile eines Anfragemodells als Bedingungen aufgefasst, die ein potentielles Ergebnismodell erfüllen muss. Auch der Ablauf der Ergebnisberechnung ist bei beiden Ansätzen ähnlich, da jeweils zunächst die Übereinstimmung von Knoten zwischen einem Anfragemodell und potentiellen Ergebnismodellen überprüft wird und erst anschließend Kanten und Pfade eines Anfragemodells auf Kanten der Ergebnismodelle abgebildet werden. Diese Schritte wirken wie ein Filter, durch den potentielle Ergebnismodelle entfernt werden, die eine Bedingung nicht erfüllen.

Im Hinblick auf die Ausdrucksmächtigkeit lassen sich mit der PNQML die gleichen Prozessmodelle wie mit Petri-Netzen ohne Kantengewichte und Marken darstellen. Zusätzlich können Pfade zwischen den Knoten eines Petri-Netzes sowie Knoten von variablem Typ beschrieben werden. Des Weiteren können die Beschriftungen von Modellelementen einen Platzhalter für eine beliebige Beschriftung enthalten.

Ein Aspekt, der hingegen nicht mit der PNQML ausgedrückt werden kann, ist das Ablaufverhalten eines solchen Anfragemodells, da die PNQML-Modelle keine Marken enthalten können. Folglich können in Anfragen keine Restriktionen für das Ablaufverhalten eines Prozessmodells beschrieben werden. Darüber hinaus werden alle in einem PNQML-Modell beschriebenen Bedingungen als durch ein logisches Und verknüpft angesehen. Eine andere Verknüpfungsart wie (exklusives) Oder ist nicht Bestandteil der PNQML. D. h., dass ein Ergebnismodell immer alle Bedingungen erfüllen muss und beispielsweise nicht ausgedrückt werden kann, dass ein Ergebnismodell eine Transition a oder eine Transition b enthalten soll.

6.5 Zusammenfassung

In diesem Kapitel wurde mit der *Petri Net Query Model Language* (PNQML) eine Anfragesprache für Prozessmodelle beschrieben. Mit dieser ist es möglich, Anfragen als ein

Modell darzustellen. Die PNQML basiert dabei auf Petri-Netzen und erweitert diese um zusätzliche Modellierungselemente, um in einer Anfrage beispielsweise Pfade zwischen Knoten oder auszuschließende Knoten spezifizieren zu können. Somit ist sie allerdings auch nur für als Petri-Netze repräsentierte Prozessmodelle einsetzbar, Modelle in anderen Modellierungssprachen können nicht abgefragt werden. Die Syntax, Semantik und Notation der PNQML wurden in Kapitel 6.3.1 erläutert. Zudem wurde in Abschnitt 6.3.2 ein Algorithmus zur Ergebnisberechnung beschrieben.

Auch im Kontext von Anfragen mit der PNQML nehmen die Beschriftungen in PNQML-Modellen wie schon bei den zuvor erörterten Suchfunktionalitäten eine zentrale Rolle ein. Die in PNQML-Modellen vorhandenen Beschriftungen von Knoten müssen identisch in potentiellen Ergebnismodellen vorliegen, was vergleichsweise einschränkend ist, da bereits die Verwendung von Synonymen Suchergebnisse verändert. Daher wurde in Kapitel 6.3.3 ein Ansatz vorgestellt, der die PNQML mit Prozessmodell-Matching verbindet. Dazu wurde diskutiert welche Anforderungen an eine solche Kombination existieren und wie diese umgesetzt werden können. Dabei wurde insbesondere auf die Auswahl von Modellelementen in PNQML-Modellen, für die Matches bei der Ergebnisberechnung berücksichtigt werden sollen, sowie Besonderheiten bei der Berücksichtigung der Struktur- und Verhaltensdimensionen erörtert. Schließlich wurde ein konkreter Vorschlag zur Kombination der PNQML mit den Triple-S Verfahren beschrieben.

In Kapitel 6.4 wurde diese Kombination mit zwei in der Literatur beschriebenen Ansätzen zur Verbindung von Prozessmodell-Anfragesprachen mit Prozessmodell-Matching verglichen. Dabei ergeben sich zusammenfassend folgende Gemeinsamkeiten und Unterschiede: Wie in (Awad u. a. 2008) für die Modellierungssprache BPMN wurde mit der PNQML eine Anfragesprache für Modelle in einer bestimmten Modellierungssprache entworfen. Zudem ist als Gemeinsamkeit zu (Yan u. a. 2012b) festzuhalten, dass die Bestandteile eines Anfragemodells als Bedingungen aufgefasst werden, die durch potentielle Ergebnismodelle erfüllt werden müssen. Darüber hinaus ist der Ablauf zur Berechnung von Anfrageergebnissen ähnlich strukturiert. Als wesentlicher Unterschied zwischen der PNQML und den beiden anderen Anfragesprachen ergibt sich hingegen die Berücksichtigung von Matches bei der Ergebnisberechnung. So wird dieser Aspekt lediglich bei der Beschreibung der PNQML ausführlicher diskutiert und ein entsprechender Vorschlag zum Einbezug von Prozessmodell-Matching Verfahren und komplexen Matches (siehe Definition 4.2) gemacht.

Kapitel 7

Implementierungsaspekte

In diesem Kapitel wird in Abschnitt 7.1 eine Architekturübersicht für eine Möglichkeit zur Integration der entwickelten Triple-S und LS3 Bibliotheken in ein Prozessmodellierungswerkzeug beschrieben. Darüber hinaus wird auch die Einbindung einer zu entwickelnden Modellierungskomponente für die PNQML erläutert. In Kapitel 7.2 wird abschließend ein erster Forschungsprototyp vorgestellt, der sich in Entwicklung befindet, um einen Einblick in eine konkrete Implementierung zu geben.

7.1 Architekturübersicht

Für eine mögliche Integration der entworfenen Suchfunktionalitäten in ein Software-Werkzeug zur Prozessmodellierung wird im Folgenden eine Web-basierte Lösung skizziert, die auf Mirco-Services zurückgreift. Abbildung 7.1 zeigt eine Übersicht der Architektur. Die grundlegende Idee dabei ist, dass einzelne Funktionalitäten des Modellierungswerkzeugs jeweils durch einen Micro-Service realisiert werden und diese Services über eine Schnittstelle miteinander kommunizieren können.

In Abbildung 7.1 sind fünf Services dargestellt, die für die Modellierung und Speicherung von Prozessmodellen sowie zur Realisierung der beschriebenen Suchmöglichkeiten benötigt werden. Dabei wird die Modellierung von Prozessen bzw. die Modellierung von Anfragen durch die PNQML jeweils durch einen Service umgesetzt, der eine grafische Oberfläche zur Darstellung bereitstellen. Die Speicherung und die Bereitstellung von Modellen findet durch den Repository-Service statt. Die Triple-S

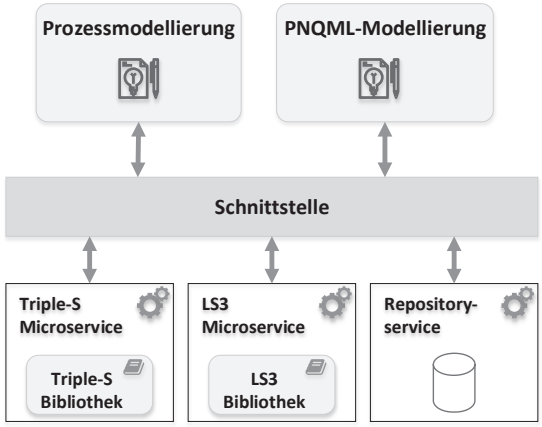


Abb. 7.1: Architekturübersicht zur Integration der Suchmöglichkeiten in ein Werkzeug zur Prozessmodellierung

und LS3 Bibliotheken können in diese Architektur eingebunden werden, indem sie ebenfalls als Micro-Services realisiert werden. Die Berechnung von Suchergebnissen findet dann durch einen Aufruf dieser Services aus den Modellierungsservices heraus statt. Beispielsweise können von dem Service zur Prozessmodellierung dem Triple-S Service zwei Modelle übergeben werden, für die ein Matching berechnet werden soll. Das Matching-Ergebnis wird anschließend an den Prozessmodellierungsservice zurückgegeben, der das Ergebnis visuell darstellen kann.

7.2 Forschungsprototyp

Am Institut für Angewandte Informatik und Formale Beschreibungsverfahren des Karlsruher Instituts für Technologie wird derzeit ein Forschungsprototyp zur Modellierung und Analyse von Geschäftsprozessen entwickelt. Dieser *PetriAnalyzer* erlaubt zum jetzigen Zeitpunkt die Modellierung von Prozessen als Petri-Netze sowie die Speicherung der Modelle im PNML-Format. Zudem kann über verschiedene Analysealgorithmen die Qualität von Prozessmodellen untersucht werden. Eine Übersicht der Architektur des *PetriAnalyzer* ist im linken Teil von Abbildung 7.2 dargestellt.

Die Modellierung von Prozessen sowie die Speicherung und Verwaltung von Modellsammlungen ist über eine Weboberfläche möglich. Das Backend bietet über RESTful

Webservices verschiedene Schnittstellen, um Modelle im PNML-Format zu importieren und zur Ausführung von Analysealgorithmen zur Berechnung der Modellqualität. Die Modelle selbst werden in einer Datenbank gespeichert, wobei eine Schnittstelle über einen RESTful Webservice einen Export als PNML-Datei ermöglicht.

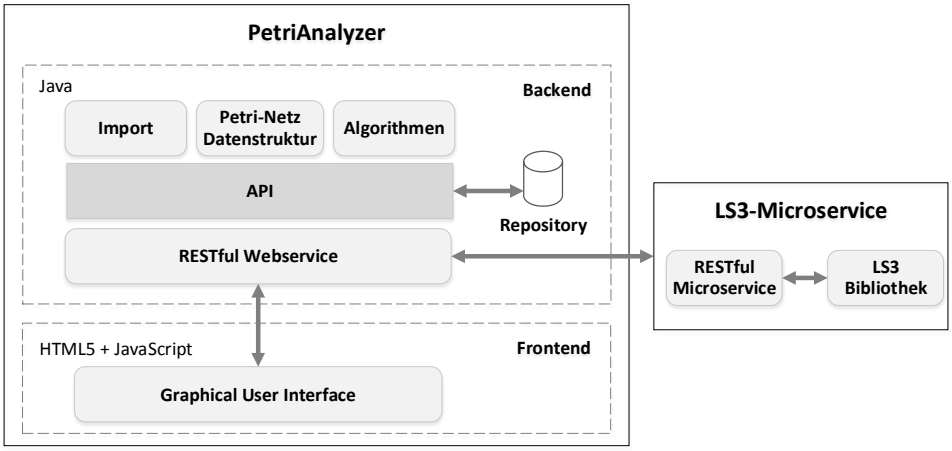


Abb. 7.2: Architektur des PetriAnalyzer Prototypen mit LS3-Microservice

In diese vorhandene Architektur wurde die existierende LS3-Bibliothek[1] über einen Microservice[2] eingebunden (rechte Seite von Abbildung 7.2). Über die Weboberfläche können dabei Modelle ausgewählt werden, zu denen ähnliche Modelle gesucht werden sollen. Der LS3-Microservice nutzt dazu die Export-Schnittstelle des *PetriAnalyzer*, um die ausgewählten Modelle aus der Datenbank zu extrahieren. Anschließend werden diese Modelle der LS3-Bibliothek übergeben, durch die für jedes Modell die ähnlichen Modelle ermittelt werden. In der momentanen Implementierung wird dazu der LS3-QueryAll Algorithmus verwendet. Anschließend werden zu jedem Modell die ähnlichen Modelle an den *PetriAnalyzer* übergeben, sodass die Anfrageergebnisse in der Weboberfläche angezeigt werden können.

Eine Darstellung der Eingabeparameter sowie der Ergebnisse einer Anfrage in der Weboberfläche sind in Abbildung 7.3 dargestellt. Oben sind dabei die Auswahl des Algorithmus und die Felder der Eingabeparameter abgebildet. Die Auswahl zu ver-

[1] Zum Download verfügbar unter https://github.com/ASchoknecht/LS3.
[2] Zum Download verfügbar unter https://github.com/cr44sh/LS3-Microservice.

gleichender Modelle ist über das Dropdown-Feld möglich. Bei der Ergebnisausgabe wird in der ersten Spalte das jeweilige Anfragemodelle angegeben, die Modelle mit einem Ähnlichkeitswert größer oder gleich θ sind in der 2. Spalte mit dem jeweiligen Ähnlichkeitswert in Klammern aufgelistet und die Parameterwerte werden schließlich in den letzten beiden Spalten dargestellt.

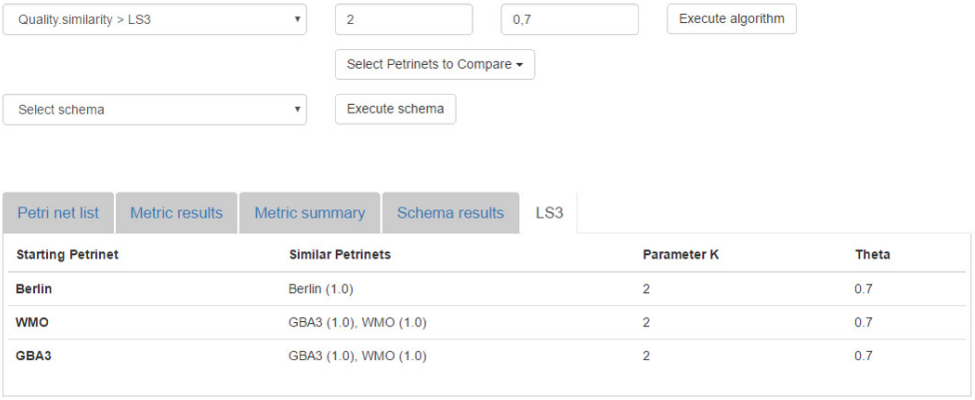


Abb. 7.3: Ansicht von Suchergebnissen des LS3-Microservice

Teil III
Schluss

Kapitel 8

Zusammenfassung und Ausblick

8.1 Zusammenfassung

Unternehmen und andere Organisationen verwalten zunehmend große Sammlungen von Geschäftsprozessmodellen. So beinhaltet die Prozessmodellbibliothek der Suncorp Gruppe, welche Bank- und Versicherungsdienstleistungen anbietet, mehr als 6.000 Prozessmodelle (Lau u. a. 2011). Auch Sammlungen von Referenzprozessmodellen können einige hundert Modelle enthalten. Das SAP Referenzmodell besteht etwa aus mehr als 600 Modellen (Curran und Ladd 1999). Um diese Mengen an Modellen sinnvoll verwalten und nutzen zu können, sind Suchfunktionalitäten nützlich, die neben einer reinen Suche mit Stichwörtern auch Modellteile oder Modelle selbst als Sucheingabe nutzen. So kann es beispielsweise sinnvoll sein, ähnliche Modellteile in einer Bibliothek zu ermitteln, um diese anschließend zu vereinheitlichen und in ein separates Modell auszulagern, wie von Uba u. a. (2011) angedacht.

Ein weiterer Anwendungsfall betrifft die Wiederverwendung von Prozessmodellen oder Modellteilen bei der Modellierung selbst. Die Modellierung von Geschäftsprozessen wird mit einem hohen Zeitaufwand sowie hohen Kosten assoziiert (vgl. etwa (Becker u. a. 2000; Becker u. a. 2010)), sodass es sinnvoll ist, existierende Modelle wiederzuverwenden, um diese Nachteile zu reduzieren. Bei der Erstellung eines neuen Modells könnten etwa existierende Teile anderer Modelle eingefügt und bei Bedarf angepasst werden.

Eine wichtige Voraussetzung für die Wiederverwendung ist allerdings, dass passende Modelle von einem Modellierer gefunden werden können. Eine manuelle Suche in hunderten von Modellen werden die wenigstens Modellierer auf sich nehmen, sodass Suchfunktionalitäten in Modellierungswerkzeugen notwendig sind. Eine rein stichwortbasierte Suchfunktion erfüllt diese Anforderung jedoch nicht. So kann eine Ergebnisliste z. B. zu viele Einträge mit passenden Modellen enthalten, wenn ein Stichwort zu unspezifisch und in vielen Modellen enthalten ist. Oder es können nur wenige passende Modelle enthalten sein, wenn statt des gesuchten Stichworts ein Synonym in den Modellen verwendet wird.

Daher wurden in dieser Ausarbeitung drei verschiedene Suchfunktionalitäten entworfen, die Prozessmodelle selbst als Sucheingabe verwenden. Die erste zielt auf die Suche nach ähnlichen Aktivitäten in Prozessmodellen ab. Diese Suchfunktionalität kann beispielsweise dazu eingesetzt werden, um die Beschriftungen oder sonstige Angaben wie Kosten, Ausführungsdauer, etc. zu diesen Aktivitäten zu vereinheitlichen. Die zweite Suchmöglichkeit kann verwendet werden, um nach ähnlichen Modellen in einer Modellbibliothek zu suchen, um diese zu vereinheitlichen oder zur Modellierung neuer Prozesse zu verwenden. Die dritte Suchfunktionalität erlaubt schließlich die Suche nach Modellen mit Hilfe einer grafischen Anfragesprache. Modellierer können damit eine Anfrage als Modell darstellen und müssen somit keine datenbankspezifischen Anfragesprachen beherrschen, um komplexere Anfragen formulieren zu können.

Bei allen Suchmöglichkeiten ist die *Ähnlichkeit* von Prozessmodellen oder Aktivitäten in diesen Modellen von zentraler Bedeutung. Bei der Suche nach Aktivitäten sollen etwa nicht nur Aktivitäten mit der gleichen Beschriftung gefunden werden, sondern auch Aktivitäten mit unterschiedlicher Beschriftung, die sich aber auf die gleiche Tätigkeit in der realen Welt beziehen. Zur Bestimmung eines Ähnlichkeitswerts können verschiedene *Dimensionen* eines Prozessmodells herangezogen werden (siehe Kapitel 2.2.2). Zu den typischerweise berücksichtigten Aspekten zählen natürlich-sprachliche Bestandteile von Modellen, die Graphstruktur und das Ablaufverhalten eines Modells. Zur Quantifizierung der Ähnlichkeit von Aktivitäten oder Prozessmodellen wird üblicherweise das Intervall $[0, 1]$ verwendet, wobei kleinere Werte eine geringere Ähnlichkeit und größere Werte eine größere Ähnlichkeit repräsentieren. Der Wert 1 kann schließlich als Gleichheit aufgefasst werden (Wombacher und Rozie 2006).

In den folgenden Absätzen werden zunächst die zentralen Ergebnisse für die Triple-S Verfahren zum Prozessmodell-Matching (Kapitel 4), für den LS3-Ansatz zur ähnlichkeitsbasierten Suche nach Prozessmodellen (Kapitel 5) sowie zur Anfragesprache PNQML (Kapitel 6) erläutert. Anschließend wird in Kapitel 8.2 ein Ausblick auf zukünftige Verbesserungsmöglichkeiten gegeben.

Prozessmodell-Matching mit Triple-S: Für das Auffinden von ähnlichen Aktivitäten, sogenannten *Matches*, in verschiedenen Prozessmodellen wurden zwei Verfahren entworfen. Beide Ansätze berechnen drei Ähnlichkeitswerte für zu vergleichende Aktivitätspaare, die schließlich zu einem finalen Ähnlichkeitswert aggregiert werden. Sollte dieser finale Wert über einem Schwellenwert liegen, werden die beiden Aktivitäten als Match aufgefasst.

Die drei berechneten Ähnlichkeitswerte werden als syntaktische, semantische und strukturelle Ähnlichkeit bezeichnet. Für die syntaktische Ähnlichkeit wird die Levenshtein-Distanz (Levenshtein 1966) der Beschriftungen von Aktivitäten berechnet. Je größer diese Distanz ist, umso unähnlicher sind sich zwei Aktivitäten hinsichtlich dieses Aspekts. Bezüglich der semantischen Ähnlichkeit werden Wortähnlichkeiten zum Vergleich der Beschriftungen genutzt. Bei dem Triple-S Verfahren wird dazu auf den Ansatz von (Wu und Palmer 1994) zurückgegriffen, während für das Triple-S2 Verfahren Wortähnlichkeiten basierend auf WORD2VEC (Mikolov u. a. 2013b) verwendet werden. Schließlich wird zur Bestimmung der strukturellen Ähnlichkeit die Position von Aktivitäten in den Prozessmodellen verglichen. Weitere Details zu den Verfahren finden sich in Kapitel 4.3.

Ergebnis einer Evaluation der beiden Triple-S Verfahren waren F-Werte im Bereich von 0,48 bis 0,57. Die Precision-Werte waren dabei vergleichsweise gut und lagen im Intervall zwischen 0,62 und 0,87, während die Recall-Werte hingegen wesentlich schlechter ausgefallen sind und lediglich im Bereich zwischen 0,39 und 0,43 lagen. Somit stellt insbesondere die Erkennung von korrekten Matches eine Herausforderung dar. Allerdings schneiden die Triple-S Verfahren im Vergleich mit zwei der besten Ansätze des Model Matching Contest 2015 (Antunes u. a. 2015) trotz der relativ geringen Werte gut ab. Insgesamt war einer der beiden verglichenen Ansätze besser und einer schlechter als die Triple-S Verfahren hinsichtlich der F-Werte. Dabei war der beste F-Werte der verglichenen Ansätze mit 0,62 nur geringfügig besser.

Bei der Evaluation wurde jedoch offensichtlich, dass die Bestimmung von korrekten Matches durch mehrere Modellierungsexperten zu sehr unterschiedlichen Goldstan-

dards führt. So wurden für den ersten Evaluationsdatensatz nur 47,8 % aller genannten Matches von mindestens drei der fünf Experten genannt. Bei dem zweiten Datensatz waren es sogar nur 29,5 %. Dadurch wird offensichtlich, dass die Festlegung korrekter Matches sich stark unterscheiden kann, was auch in einer weiteren Untersuchung festgestellt wurde (Rodriguez u. a. 2016). Wenn bei der Evaluation nur die Matches als korrekt angesehen werden, die von mindestens drei Experten genannt werden, schneiden die Triple-S Verfahren deutlich besser ab. Die F-Werte steigen auf den Bereich von 0,54 bis 0,74, wobei nach wie vor die Precision-Werte größer sind als die Recall-Werte. Somit wird allerdings ersichtlich, dass die Triple-S Verfahren die als eher eindeutig anzusehenden Matches vergleichsweise gut erkennen.

Ähnlichkeitsbasierte Suche mit LS3: Zur Suche nach ähnlichen Modellen in einer Prozessmodellbibliothek wurde in Kapitel 5 das *Latent Semantic Analysis-based Similarity Search* (LS3) Verfahren beschrieben. Dabei wird auf die textuellen Beschriftungen in Modellen zur Ähnlichkeitsberechnung fokussiert, indem ein Prozessmodell im Wesentlichen als Textdokument aufgefasst wird. Andere Dimensionen wie die Graphstruktur oder das Verhalten werden hingegen nicht berücksichtigt. Die textuellen Beschriftungen in Prozessmodellen werden mit Hilfe der Latent Semantic Analysis (Deerwester u. a. 1990), einem Verfahren aus dem Information Retrieval, in eine Vektorrepräsentation transformiert. Diese erlaubt eine Ähnlichkeitsberechnung von Prozessmodellen, wozu bei dem LS3-Verfahren die Kosinus-Ähnlichkeit (van Rijsbergen 1979, S. 25) verwendet wird. Zwei Prozessmodelle sind damit umso ähnlicher je kleiner der Winkel zwischen ihren Vektordarstellungen ist. Im Vergleich zu anderen ähnlichkeitsbasierten Suchverfahren werden damit die Schwierigkeiten bei der Berechnung von Matches zwischen Prozessmodellen umgangen.

Mit den LS3-Verfahren können zum einen zu einem Anfragemodell ähnliche Modelle in einer Prozessmodellbibliothek gefunden werden. Zum anderen können direkt alle zueinander ähnlichen Modelle in einer Bibliothek ermittelt werden, d. h. jedes Modell wird als Anfrage verwendet und die dazu ähnlichen Modelle berechnet. Zudem wurden Algorithmen entworfen, um die dem LS3-Verfahren zugrunde liegende Datenstruktur anzupassen, wenn Modelle in der Bibliothek geändert oder gelöscht bzw. neue Modelle hinzugefügt werden.

Bei der Evaluation schnitt das LS3-Verfahren im Vergleich mit fünf anderen Matching-basierten Suchverfahren am besten ab. So erzielte das LS3-Verfahren die höchsten

Werte für Precision (0,98), Recall (0,88) und F-Wert (0,93). Das beste Vergleichsverfahren erreichte lediglich einen F-Wert von 0,78. Und auch bezüglich R-Precision und Precision-at-5 erzielte das LS3-Verfahren die besten Werte mit 0,96 und 0,99. Die Differenz zu den Matching-basierten Verfahren war jedoch bezüglich der letzten beiden Kennzahlen wesentlich geringer. Insbesondere ein Verfahren erzielte nahezu gleich gute Werte mit 0,93 für R-Precision und 0,98 für Precision-at-5. Schließlich war die Berechnungszeit der Suchergebnisse bei dem LS3-Verfahren wesentlich geringer als bei den Vergleichsverfahren.

Anfragen mit PNQML: Zudem wurde in Kapitel 6 mit der *Petri Net Query Model Language* (PNQML) eine Anfragesprache für Petri-Netz basierte Prozessmodelle beschrieben. Mit Hilfe dieser Sprache können Anfragen durch ein Modell ausgedrückt werden, wozu eine Notation verwendet werden kann, die an die gängige Petri-Netz Notation angelehnt ist. Zusätzlich wird sie um weitere Elemente ergänzt, um z. B. Pfade zwischen Knoten darstellen zu können. Die Kernidee der PNQML ist, dass durch jedes in einem PNQML-Modell verwendete Element eine Bedingung formuliert wird, die die gesuchten Modelle erfüllen müssen, um in das Suchergebnis aufgenommen zu werden. Dazu wurde ein Algorithmus vorgestellt, der ein PNQML-Modell als Anfrage enthält und durch den Ergebnismodelle in einer Prozessmodellbibliothek bestimmt werden können.

Des Weiteren wurde die Integration von Prozessmodell-Matching Ansätzen wie beispielsweise den in Kapitel 4 erläuterten Triple-S Verfahren in die PNQML diskutiert. Ein zentraler Aspekt sind notwendige Anpassungen von Matching-Verfahren, da etwa die Struktur- und Verhaltensdimensionen nicht wie beim Matching von zwei Prozessmodellen berücksichtigt werden können. Zudem wurden die Spezifizierung von Elementen in PNQML-Modellen, für die Matches bei der Ergebnisberechnung berücksichtigt werden sollen, sowie eine Anpassung des Algorithmus zur Ergebnisberechnung beschrieben.

Durch die Diskussion dieser Aspekte wurde insbesondere ein Beitrag dazu geliefert, wie Anfragesprachen für Prozessmodelle um Prozessmodell-Matching erweitert werden können und welche Problemstellungen sich dadurch ergeben. Verwandte Arbeiten versuchen anhand von Ähnlichkeitsberechnungen zwischen einem Anfragemodell und potentiellen Ergebnismodellen weitere relevante Modelle zu ermitteln, die beispielsweise geringfügig bezüglich der Graphstruktur abweichen oder die ähnliche

Wörter in den Beschriftungen verwenden. Eine explizite Kennzeichnung von Elementen in einem Anfragemodell, für die Matches berechnet werden sollen, wurde allerdings bislang nicht erläutert.

8.2 Ausblick

Prozessmodell-Matching: Zur Verbesserung der Triple-S Verfahren könnten einige Heuristiken integriert werden, die möglicherweise Lösungsmöglichkeiten für die in Kapitel 4.4.3 erläuterten Problemklassen bieten. Beispielsweise könnte die Regel eingeführt werden, dass gleich beschriftete Aktivitäten unabhängig von der strukturellen Ähnlichkeit ein Match ergeben. Eine weitere Verbesserung der Matching-Ergebnisse könnte durch eine Berücksichtigung weiterer Informationen erzielt werden. Bislang berücksichtigen die Triple-S Verfahren etwa keine organisatorischen Angaben zu ausführenden Rollen von Aktivitäten oder in Prozessmodellen beschriebenen Ressourcen und Datenobjekten.

Die grundlegende Schwierigkeit für Matching-Ansätze, dass Beschriftungen typischerweise lediglich wenige Wörter enthalten (Koschmider u. a. 2015) und auf unterschiedliche Weise modelliert werden können, könnte durch zusätzliche Aktivitätsbeschreibungen verringert werden. Solche zusätzlichen Angaben werden bereits in Anfragen nach Prozessmodellen verwendet (Leopold u. a. 2016) und enthalten weitere Informationen zu der auszuführenden Aktivität. Durch die zusätzlichen Angaben in den Beschreibungen könnte potentiell besser erkannt werden, welche Aktivitäten sich entsprechen. Zudem könnte die Definition von Matches erweitert werden, sodass nicht nur als Petri-Netze modellierte Prozessmodelle umfasst werden, sondern auch weitere Modellierungssprachen. Dadurch könnten die Triple-S Verfahren auch für Modellsammlungen mit Modellen in unterschiedlichen Sprachen eingesetzt werden, was bislang nicht möglich ist.

Darüber hinaus sollten weitere empirische Untersuchungen und Vergleiche von Matching-Verfahren durchgeführt werden. Bislang wurden verschiedene Ansätze im Wesentlichen in den Matching Wettbewerben miteinander verglichen, wobei die Größe der Modelldatensätze klein war und lediglich ein Goldstandard verwendet wurde. Zudem wurde bislang nicht evaluiert, wie gut die Verfahren abschneiden, wenn unterschiedliche Prozesse zugrunde liegen und sich lediglich einige Aktivitäten entspre-

chen. Bislang wurden die Ergebnisse nur für mehrere Modelle eines gemeinsamen Prozesses untersucht.

Ähnlichkeitsbasierte Suche nach Prozessmodellen: Hinsichtlich zukünftiger Weiterentwicklungen des LS3-Verfahrens ist eine Erweiterung für Prozessmodelle in anderen Modellierungssprachen wie EPK (Keller u. a. 1992) oder BPMN (*Business Process Model and Notation (BPMN)* 2011) denkbar. Eine solche Erweiterung ist leicht realisierbar, da das LS3-Verfahren nur die textuellen Beschriftungen verwendet. Dies würde es ermöglichen, ähnliche Modelle, die in verschiedenen Modellierungssprachen erstellt wurden, zu ermitteln.

Darüber hinaus ist auch eine Erweiterung der Ähnlichkeitsberechnung auf Modelle in unterschiedlichen natürlichen Sprachen denkbar. Die existierende prototypische Implementierung erlaubt momentan nur eine Suche über Modellen mit englischen Beschriftungen. Dazu müssten die eingesetzten Verfahren zur Verarbeitung natürlicher Sprache angepasst werden. Die eingesetzte Stoppwort-Liste müsste etwa je nach Sprache erweitert werden, sodass z. B. auch deutsche Stoppwörter in deutschen Modellen entfernt würden. Zudem müsste ein Verfahren eingesetzt werden, um Wortstämme in der jeweiligen Sprache zu bestimmen.

Außerdem werden bislang Modelle für die Ähnlichkeitsberechnung unabhängig von hierarchischen Beziehungen zwischen den Modellen betrachtet. Dass beispielsweise ein Modell einen Teilprozess in einem anderen Modell darstellt, wird noch nicht berücksichtigt und die Eignung des LS3-Verfahrens in solchen Fällen auch nicht im Rahmen der Evaluation überprüft. Eine Lösungsmöglichkeit wäre die Repräsentation aller Modelle, die zur Darstellung eines Prozesses verwendet werden, als ein gemeinsamer Dokumentenvektor. Dieser könnte dann zur Ähnlichkeitsberechnung verwendet und die zugehörigen Modelle als Ergebnis ausgegeben werden. Hierzu müsste das LS3-Verfahren um eine Funktionalität zur Erkennung solcher gemeinsamer Dokumentenvektoren erweitert sowie die Ähnlichkeitsberechnung entsprechend angepasst werden. Bislang ist die Berücksichtigung von Modellbeziehungen dieser Art bei der Ähnlichkeitsberechnung auch in der Literatur noch nicht geklärt (Schoknecht u. a. 2017b).

Als Alternativen zur Berechnung der Ähnlichkeit von Prozessmodellen könnten anstatt der Latent Semantic Analysis die Latent Dirichlet Allocation (LDA) (Blei u. a. 2003) oder WORD2VEC (Mikolov u. a. 2013b) eingesetzt werden, mit denen im Bereich der Computerlinguistik bessere Ergebnisse erzielt werden (Baroni u. a. 2014).

Bei der Verwendung der LDA zur Berechnung der Ähnlichkeit von Prozessmodellen in (Qiao u. a. 2011) konnten die Evaluationsergebnisse jedoch nicht überzeugen. Des Weiteren könnte das LS3-Verfahren mit einem Matching-basierten Ansatz kombiniert werden, um neben der Suche nach Prozessmodellen weitere Anwendungsfälle von Ähnlichkeitsberechnungen abzudecken, für die Matches erforderlich sind (siehe Kapitel 2.2.3). Dabei könnte das LS3-Verfahren eine Art Vorverarbeitung darstellen, um die grundsätzlich ähnlichen Modelle zu ermitteln, die anschließend von einem Matching-basierten Ansatz weiter analysiert werden.

Schließlich sollten weitere Evaluationen des LS3-Verfahrens und anderer Ansätze zur Ähnlichkeitsberechnung durchgeführt werden. Zum einen sollten größere Modellsammlungen verwendet werden, um die Qualität auch im Falle von einigen hundert oder tausend Modellen zu überprüfen. Zum anderen sollte die Reihenfolge der Modelle in einem Suchergebnis detaillierter betrachtet werden. Hierbei könnten etwa automatisch berechnete Ergebnislisten mit von Menschen erzeugten Reihenfolgen verglichen werden, um zu überprüfen, inwieweit die automatischen Ergebnisse mit der menschlichen Einschätzung übereinstimmen.

Anfragesprachen für Prozessmodellbibliotheken: Zur Untersuchung und Bewertung der PNQML im praktischen Einsatz ist eine Implementierung eines Editors für die Modellierung von PNQML-Modellen notwendig. Dies könnte wie in Kapitel 7 beschrieben Web-basiert erfolgen. Dadurch könnten beispielsweise empirische Vergleiche zwischen Anfragen mit PNQML-Modellen und einer manuellen Suche nach Modellen durch Browsen in einer Modellbibliothek und mit Hilfe von Stichwörtern durchgeführt werden, um die Nützlichkeit der PNQML zu bewerten. Auch könnten durch eine Implementierung die benötigte Zeit zur Berechnung von Anfrageergebnissen bestimmt und gegebenenfalls Indexierungsstrukturen entworfen werden.

Des Weiteren sollten verwandte Anfragesprachen zur Suche nach Prozessmodellen und die PNQML miteinander verglichen werden. Dabei sollte insbesondere auch der Einbezug von Matches bei der Ergebnisberechnung untersucht werden, um zu bestimmen, inwieweit diese tatsächlich hilfreich sind bzw. wie der in Kapitel 6.3.3 erläuterte Ansatz im Vergleich zu anderen abschneidet.

Weitere zukünftig sinnvolle Forschungsarbeiten im Bereich der (ähnlichkeitsbasierten) Suche in Prozessmodellbibliotheken wären zusätzliche Vergleiche von Suchverfahren, wie sie etwas durch die Prozessmodell-Matching Wettbewerbe (Antunes u. a.

2015; Cayoglu u. a. 2014) angestoßen wurden. Darüber hinaus sind empirische Untersuchungen wie z. B. von Thaler u. a. (2017) zur Korrelation von Ähnlichkeitswerten verschiedener Verfahren hilfreich. Dadurch könnte geklärt werden, welche Suchverfahren sich für welche Einsatzszenarien eignen, was bislang noch eine offene Fragestellung ist (Schoknecht u. a. 2017b).

Des Weiteren ist die tatsächliche Wiederverwendung von Prozessmodellen durch Suchfunktionalitäten im praktischen Einsatz von Modellierern von Interesse. Die Evaluationen in dieser Arbeit beziehen sich im Wesentlichen auf technische Aspekte und einen Vergleich der Güte unterschiedlicher Verfahren. Eine Aussage über die praktische Nützlichkeit für Modellierer kann dadurch jedoch nicht abschließend getroffen werden und bleibt damit für zukünftige Untersuchungen offen. Schließlich könnte auch der Einsatz von ähnlichkeitsbasierten Suchfunktionen in anderen Anwendungsbereichen sinnvoll sein. So wurde beispielsweise von Fellmann u. a. (2016) untersucht, inwieweit sich Prozessmodell-Matching Verfahren zur automatischen Bewertung und Korrektur von Modellierungsaufgaben im Hochschulkontext einsetzen lassen. Hier lassen sich sicherlich noch weitere interessante Einsatzmöglichkeiten finden.

Anhang A

Anhang

A.1 Prozessmodell-Matching Ergebnisse

Die detaillierten Ergebnisse der Triple-S und Triple-S2 Verfahren für die einzelnen Goldstandards sind in den folgenden Tabellen abgebildet. Die Ergebnisse für die Zulassungsmodelle (University Admisison) sind in den Tabellen A.1 bis A.7 aufgelistet, während sich die Ergebnisse für die Geburtsregistrierungsmodelle (Birth Registration) in den Tabellen A.8 bis A.14 finden. Als Parameterkombination der beiden Verfahren wurde jeweils eine Kombination gewählt, die den Micro-Average F-Wert der Zulassungsmodelle maximiert. Für den Triple-S Ansatz wurden folgende Parameterwerte ermittelt: $\omega_1 = 0,28$, $\omega_2 = 0,27$, $\omega_3 = 0,05$, $\omega_4 = 0,4$, $\theta = 0,77$. Für den Triple-S2 Ansatz wurden folgende Parameterwerte ermittelt: $\omega_1 = 0,55$, $\omega_2 = 0,45$, $\omega_3 = 0,6$, $\omega_4 = 0,4$, $\theta_1 = 0,75$, $\theta_2 = 0,82$.

Tab. A.1: University Admission Matching-Ergebnisse Goldstandard 1

University Admission	Triple-S			Triple-S2		
	Precision	Recall	F-Wert	Precision	Recall	F-Wert
Cologne - Frankfurt	0,58	0,78	0,67	0,70	0,78	0,74
Cologne - FU Berlin	1,00	0,07	0,13	1,00	0,13	0,24
Cologne - Hohenheim	0,00	0,00	0,00	0,33	0,09	0,14
Cologne - IIS Erlangen	0,67	0,13	0,22	1,00	0,13	0,24
Cologne - Muenster	0,67	0,15	0,25	0,00	0,00	0,00
Cologne - Potsdam	1,00	0,18	0,31	1,00	0,18	0,31
Cologne - TU Munich	0,63	0,83	0,71	0,71	0,83	0,77
Cologne - Wuerzburg	1,00	0,13	0,22	0,50	0,13	0,20
Frankfurt - FU Berlin	1,00	0,22	0,36	1,00	0,22	0,36
Frankfurt - Hohenheim	0,00	0,00	0,00	1,00	0,27	0,43
Frankfurt - IIS Erlangen	1,00	0,00	0,00	1,00	0,18	0,31
Frankfurt - Muenster	0,00	0,00	0,00	1,00	0,10	0,18
Frankfurt - Potsdam	1,00	0,00	0,00	1,00	0,09	0,17
Frankfurt - TU Munich	0,53	1,00	0,69	1,00	1,56	1,22
Frankfurt - Wuerzburg	1,00	0,13	0,22	1,00	0,25	0,40
FU Berlin - Hohenheim	1,00	0,00	0,00	1,00	0,11	0,20
FU Berlin - IIS Erlangen	0,58	0,82	0,68	0,58	0,82	0,68
FU Berlin - Muenster	0,35	0,40	0,38	0,50	0,40	0,44
FU Berlin - Potsdam	0,63	0,94	0,75	0,63	0,94	0,75
FU Berlin - TU Munich	1,00	0,27	0,43	1,00	0,27	0,43
FU Berlin - Wuerzburg	0,56	0,63	0,59	0,60	0,75	0,67
Hohenheim - IIS Erlangen	0,50	0,24	0,32	0,43	0,18	0,25
Hohenheim - Muenster	0,56	0,28	0,37	0,46	0,33	0,39
Hohenheim - Potsdam	0,25	0,06	0,09	0,50	0,06	0,10
Hohenheim - TU Munich	0,00	0,00	0,00	0,50	0,15	0,24
Hohenheim - Wuerzburg	1,00	0,00	0,00	0,25	0,11	0,15
IIS Erlangen - Muenster	0,30	0,36	0,33	0,36	0,36	0,36
IIS Erlangen - Potsdam	0,67	0,67	0,67	0,68	0,77	0,72
IIS Erlangen - TU Munich	0,75	0,33	0,46	0,80	0,22	0,35
IIS Erlangen - Wuerzburg	0,44	0,57	0,50	0,63	0,71	0,67
Muenster - Potsdam	0,40	0,38	0,39	0,42	0,38	0,40
Muenster - TU Munich	0,63	0,31	0,42	0,67	0,25	0,36
Muenster - Wuerzburg	0,30	0,20	0,24	0,33	0,13	0,19
Potsdam - TU Munich	1,00	0,31	0,47	1,00	0,31	0,47
Potsdam - Wuerzburg	0,33	0,43	0,38	0,44	0,57	0,50
TU Munich - Wuerzburg	1,00	0,30	0,46	1,00	0,30	0,46
Macro Average	*0,62*	*0,31*	*0,33*	*0,70*	*0,36*	*0,40*
Standardabweichung	*0,33*	*0,29*	*0,24*	*0,28*	*0,33*	*0,24*
Micro Average	*0,52*	*0,33*	*0,40*	*0,61*	*0,37*	*0,46*

Tab. A.2: University Admission Matching-Ergebnisse Goldstandard 2

University Admission	**Triple-S**			**Triple-S2**		
	Precision	Recall	F-Wert	Precision	Recall	F-Wert
Cologne - Frankfurt	0,58	1,00	0,74	0,70	1,00	0,82
Cologne - FU Berlin	1,00	0,25	0,40	1,00	0,50	0,67
Cologne - Hohenheim	0,00	0,00	0,00	0,33	0,33	0,33
Cologne - IIS Erlangen	0,00	0,00	0,00	0,00	0,00	0,00
Cologne - Muenster	0,00	0,00	0,00	0,00	0,00	0,00
Cologne - Potsdam	0,00	0,00	0,00	0,00	0,00	0,00
Cologne - TU Munich	0,44	1,00	0,61	0,50	1,00	0,67
Cologne - Wuerzburg	0,00	0,00	0,00	0,00	0,00	0,00
Frankfurt - FU Berlin	0,00	0,00	0,00	0,00	0,00	0,00
Frankfurt - Hohenheim	0,00	0,00	0,00	0,33	1,00	0,50
Frankfurt - IIS Erlangen	1,00	0,00	0,00	0,00	0,00	0,00
Frankfurt - Muenster	0,00	0,00	0,00	0,00	0,00	0,00
Frankfurt - Potsdam	1,00	0,00	0,00	0,00	0,00	0,00
Frankfurt - TU Munich	0,53	1,00	0,69	0,64	1,00	0,78
Frankfurt - Wuerzburg	0,00	0,00	0,00	0,00	0,00	0,00
FU Berlin - Hohenheim	1,00	0,00	0,00	0,00	0,00	0,00
FU Berlin - IIS Erlangen	0,50	0,92	0,65	0,50	0,92	0,65
FU Berlin - Muenster	0,29	0,56	0,38	0,42	0,56	0,48
FU Berlin - Potsdam	0,58	1,00	0,74	0,58	1,00	0,74
FU Berlin - TU Munich	0,00	0,00	0,00	0,00	0,00	0,00
FU Berlin - Wuerzburg	0,44	0,67	0,53	0,50	0,83	0,63
Hohenheim - IIS Erlangen	0,13	0,20	0,15	0,14	0,20	0,17
Hohenheim - Muenster	0,33	0,38	0,35	0,31	0,50	0,38
Hohenheim - Potsdam	0,25	0,17	0,20	0,50	0,17	0,25
Hohenheim - TU Munich	0,00	0,00	0,00	0,25	0,20	0,22
Hohenheim - Wuerzburg	1,00	0,00	0,00	0,25	0,33	0,29
IIS Erlangen - Muenster	0,24	0,62	0,35	0,29	0,62	0,39
IIS Erlangen - Potsdam	0,60	0,82	0,69	0,59	0,91	0,71
IIS Erlangen - TU Munich	0,25	0,25	0,25	0,00	0,00	0,00
IIS Erlangen - Wuerzburg	0,33	0,43	0,38	0,50	0,57	0,53
Muenster - Potsdam	0,35	0,64	0,45	0,37	0,64	0,47
Muenster - TU Munich	0,13	0,17	0,14	0,00	0,00	0,00
Muenster - Wuerzburg	0,30	0,50	0,38	0,33	0,33	0,33
Potsdam - TU Munich	0,00	0,00	0,00	0,00	0,00	0,00
Potsdam - Wuerzburg	0,33	0,50	0,40	0,44	0,67	0,53
TU Munich - Wuerzburg	0,00	0,00	0,00	0,00	0,00	0,00
Macro Average	*0,32*	*0,31*	*0,24*	*0,26*	*0,37*	*0,29*
Standardabweichung	*0,34*	*0,36*	*0,26*	*0,26*	*0,38*	*0,29*
Micro Average	*0,36*	*0,51*	*0,43*	*0,40*	*0,55*	*0,46*

Tab. A.3: University Admission Matching-Ergebnisse Goldstandard 3

University Admission	Triple-S			Triple-S2		
	Precision	Recall	F-Wert	Precision	Recall	F-Wert
Cologne - Frankfurt	0,58	0,70	0,64	0,70	0,70	0,70
Cologne - FU Berlin	1,00	0,09	0,17	1,00	0,18	0,31
Cologne - Hohenheim	0,00	0,00	0,00	0,33	0,07	0,12
Cologne - IIS Erlangen	0,67	0,17	0,27	1,00	0,17	0,29
Cologne - Muenster	0,67	0,09	0,16	1,00	0,09	0,17
Cologne - Potsdam	1,00	0,40	0,57	1,00	0,40	0,57
Cologne - TU Munich	0,56	1,00	0,72	0,64	1,00	0,78
Cologne - Wuerzburg	1,00	0,14	0,25	0,50	0,14	0,22
Frankfurt - FU Berlin	1,00	0,29	0,44	1,00	0,29	0,44
Frankfurt - Hohenheim	0,00	0,00	0,00	0,33	0,11	0,17
Frankfurt - IIS Erlangen	1,00	0,00	0,00	1,00	0,40	0,57
Frankfurt - Muenster	0,00	0,00	0,00	1,00	0,06	0,11
Frankfurt - Potsdam	1,00	0,00	0,00	1,00	0,20	0,33
Frankfurt - TU Munich	0,41	1,00	0,58	0,50	1,00	0,67
Frankfurt - Wuerzburg	1,00	0,20	0,33	0,50	0,20	0,29
FU Berlin - Hohenheim	1,00	0,00	0,00	1,00	0,13	0,22
FU Berlin - IIS Erlangen	0,58	0,82	0,68	0,58	0,82	0,68
FU Berlin - Muenster	0,29	0,50	0,37	0,42	0,50	0,45
FU Berlin - Potsdam	0,63	1,00	0,77	0,63	1,00	0,77
FU Berlin - TU Munich	1,00	0,38	0,55	1,00	0,38	0,55
FU Berlin - Wuerzburg	0,44	0,50	0,47	0,60	0,75	0,67
Hohenheim - IIS Erlangen	0,38	0,23	0,29	0,29	0,15	0,20
Hohenheim - Muenster	0,56	0,31	0,40	0,46	0,38	0,41
Hohenheim - Potsdam	0,25	0,11	0,15	0,50	0,11	0,18
Hohenheim - TU Munich	0,00	0,00	0,00	0,25	0,09	0,13
Hohenheim - Wuerzburg	1,00	0,00	0,00	0,50	0,22	0,31
IIS Erlangen - Muenster	0,33	0,52	0,41	0,39	0,52	0,45
IIS Erlangen - Potsdam	0,63	0,95	0,76	0,56	0,95	0,70
IIS Erlangen - TU Munich	0,75	0,43	0,55	0,80	0,29	0,42
IIS Erlangen - Wuerzburg	0,56	0,56	0,56	0,75	0,67	0,71
Muenster - Potsdam	0,35	0,64	0,45	0,37	0,64	0,47
Muenster - TU Munich	0,63	0,20	0,30	0,67	0,16	0,26
Muenster - Wuerzburg	0,30	0,30	0,30	0,33	0,20	0,25
Potsdam - TU Munich	1,00	0,50	0,67	1,00	0,50	0,67
Potsdam - Wuerzburg	0,44	0,44	0,44	0,67	0,67	0,67
TU Munich - Wuerzburg	0,67	0,22	0,00	0,67	0,22	0,33
Macro Average	*0,60*	*0,35*	*0,34*	*0,66*	*0,40*	*0,42*
Standardabweichung	*0,33*	*0,31*	*0,25*	*0,25*	*0,30*	*0,21*
Micro Average	*0,50*	*0,37*	*0,43*	*0,56*	*0,40*	*0,47*

Tab. A.4: University Admission Matching-Ergebnisse Goldstandard 4

University Admission	**Triple-S**			**Triple-S2**		
	Precision	Recall	F-Wert	Precision	Recall	F-Wert
Cologne - Frankfurt	0,42	1,00	0,59	0,50	1,00	0,67
Cologne - FU Berlin	1,00	0,13	0,22	1,00	0,25	0,40
Cologne - Hohenheim	0,50	0,13	0,20	0,33	0,13	0,18
Cologne - IIS Erlangen	0,67	0,33	0,44	1,00	0,33	0,50
Cologne - Muenster	0,67	0,33	0,44	1,00	0,33	0,50
Cologne - Potsdam	1,00	0,20	0,33	1,00	0,20	0,33
Cologne - TU Munich	0,50	1,00	0,67	0,57	1,00	0,73
Cologne - Wuerzburg	1,00	0,20	0,33	0,50	0,20	0,29
Frankfurt - FU Berlin	1,00	0,29	0,44	1,00	0,29	0,44
Frankfurt - Hohenheim	0,00	0,00	0,00	0,33	0,17	0,22
Frankfurt - IIS Erlangen	1,00	0,00	0,00	1,00	0,29	0,44
Frankfurt - Muenster	0,00	0,00	0,00	1,00	0,25	0,40
Frankfurt - Potsdam	1,00	0,00	0,00	1,00	0,17	0,29
Frankfurt - TU Munich	0,47	1,00	0,64	0,57	1,00	0,73
Frankfurt - Wuerzburg	1,00	0,25	0,40	0,50	0,25	0,33
FU Berlin - Hohenheim	1,00	0,00	0,00	1,00	0,07	0,13
FU Berlin - IIS Erlangen	0,42	1,00	0,59	0,42	1,00	0,59
FU Berlin - Muenster	0,24	0,50	0,32	0,33	0,50	0,40
FU Berlin - Potsdam	0,63	0,94	0,75	0,63	0,94	0,75
FU Berlin - TU Munich	0,33	0,11	0,17	0,33	0,11	0,17
FU Berlin - Wuerzburg	0,22	0,50	0,31	0,30	0,75	0,43
Hohenheim - IIS Erlangen	0,50	0,29	0,36	0,29	0,14	0,19
Hohenheim - Muenster	0,44	0,27	0,33	0,54	0,47	0,50
Hohenheim - Potsdam	0,25	0,08	0,12	0,50	0,08	0,13
Hohenheim - TU Munich	0,00	0,00	0,00	0,00	0,00	0,00
Hohenheim - Wuerzburg	1,00	0,00	0,00	0,50	0,25	0,33
IIS Erlangen - Muenster	0,27	0,35	0,31	0,32	0,35	0,33
IIS Erlangen - Potsdam	0,60	0,62	0,61	0,65	0,76	0,70
IIS Erlangen - TU Munich	0,38	0,20	0,26	0,40	0,13	0,20
IIS Erlangen - Wuerzburg	0,33	0,60	0,43	0,38	0,60	0,46
Muenster - Potsdam	0,35	0,32	0,33	0,37	0,32	0,34
Muenster - TU Munich	0,50	0,36	0,42	0,50	0,27	0,35
Muenster - Wuerzburg	0,30	0,43	0,35	0,33	0,29	0,31
Potsdam - TU Munich	0,50	0,33	0,40	0,50	0,33	0,40
Potsdam - Wuerzburg	0,33	0,43	0,38	0,56	0,71	0,63
TU Munich - Wuerzburg	0,33	0,25	0,00	0,33	0,25	0,29
Macro Average	*0,53*	*0,34*	*0,31*	*0,57*	*0,39*	*0,39*
Standardabweichung	*0,31*	*0,31*	*0,21*	*0,27*	*0,30*	*0,18*
Micro Average	*0,42*	*0,37*	*0,39*	*0,49*	*0,41*	*0,44*

Tab. A.5: University Admission Matching-Ergebnisse Goldstandard 5

University Admission	Triple-S			Triple-S2		
	Precision	Recall	F-Wert	Precision	Recall	F-Wert
Cologne - Frankfurt	0,67	1,00	0,80	0,80	1,00	0,89
Cologne - FU Berlin	1,00	0,06	0,11	1,00	0,12	0,21
Cologne - Hohenheim	0,00	0,00	0,00	0,00	0,00	0,00
Cologne - IIS Erlangen	0,67	0,40	0,50	1,00	0,40	0,57
Cologne - Muenster	0,67	0,40	0,50	1,00	0,40	0,57
Cologne - Potsdam	1,00	0,33	0,50	1,00	0,33	0,50
Cologne - TU Munich	0,63	1,00	0,77	0,71	1,00	0,83
Cologne - Wuerzburg	1,00	0,09	0,17	0,50	0,09	0,15
Frankfurt - FU Berlin	1,00	0,40	0,57	1,00	0,40	0,57
Frankfurt - Hohenheim	0,00	0,00	0,00	0,33	0,11	0,17
Frankfurt - IIS Erlangen	1,00	0,00	0,00	1,00	0,33	0,50
Frankfurt - Muenster	0,00	0,00	0,00	1,00	0,17	0,29
Frankfurt - Potsdam	1,00	0,00	0,00	1,00	0,20	0,33
Frankfurt - TU Munich	0,41	1,00	0,58	0,50	1,00	0,67
Frankfurt - Wuerzburg	1,00	0,20	0,33	0,50	0,20	0,29
FU Berlin - Hohenheim	1,00	0,00	0,00	1,00	0,25	0,40
FU Berlin - IIS Erlangen	0,58	0,93	0,72	0,58	0,93	0,72
FU Berlin - Muenster	0,29	0,50	0,37	0,42	0,50	0,45
FU Berlin - Potsdam	0,58	0,93	0,72	0,58	0,93	0,72
FU Berlin - TU Munich	0,67	0,22	0,33	0,67	0,22	0,33
FU Berlin - Wuerzburg	0,44	0,80	0,57	0,50	1,00	0,67
Hohenheim - IIS Erlangen	0,13	0,33	0,18	0,14	0,33	0,20
Hohenheim - Muenster	0,22	0,40	0,29	0,23	0,60	0,33
Hohenheim - Potsdam	0,25	0,25	0,25	0,50	0,25	0,33
Hohenheim - TU Munich	0,00	0,00	0,00	0,25	0,33	0,29
Hohenheim - Wuerzburg	1,00	0,00	0,00	0,00	0,00	0,00
IIS Erlangen - Muenster	0,30	0,83	0,44	0,36	0,83	0,50
IIS Erlangen - Potsdam	0,60	0,82	0,69	0,59	0,91	0,71
IIS Erlangen - TU Munich	0,38	0,50	0,43	0,40	0,33	0,36
IIS Erlangen - Wuerzburg	0,33	0,60	0,43	0,38	0,60	0,46
Muenster - Potsdam	0,25	0,56	0,34	0,26	0,56	0,36
Muenster - TU Munich	0,38	0,30	0,33	0,50	0,30	0,38
Muenster - Wuerzburg	0,30	0,60	0,40	0,33	0,40	0,36
Potsdam - TU Munich	0,00	0,00	0,00	0,00	0,00	0,00
Potsdam - Wuerzburg	0,33	0,43	0,38	0,56	0,71	0,63
TU Munich - Wuerzburg	0,00	0,00	0,00	0,00	0,00	0,00
Macro Average	*0,50*	*0,39*	*0,33*	*0,54*	*0,44*	*0,41*
Standardabweichung	*0,35*	*0,34*	*0,26*	*0,32*	*0,32*	*0,23*
Micro Average	*0,42*	*0,47*	*0,45*	*0,48*	*0,51*	*0,50*

Tab. A.6: University Admission Matching-Ergebnisse Goldstandard Gesamt

University Admission	**Triple-S**			**Triple-S2**		
	ProP	ProR	ProFW	ProP	ProR	ProFW
Cologne - Frankfurt	0,74	0,87	0,80	0,85	0,87	0,86
Cologne - FU Berlin	1,00	0,09	0,17	1,00	0,18	0,31
Cologne - Hohenheim	0,25	0,02	0,04	0,57	0,09	0,16
Cologne - IIS Erlangen	0,73	0,20	0,31	1,00	0,20	0,33
Cologne - Muenster	0,73	0,18	0,29	1,00	0,18	0,30
Cologne - Potsdam	1,00	0,24	0,38	1,00	0,24	0,38
Cologne - TU Munich	0,71	0,96	0,81	0,79	0,96	0,86
Cologne - Wuerzburg	1,00	0,12	0,22	0,57	0,12	0,20
Frankfurt - FU Berlin	1,00	0,27	0,42	1,00	0,27	0,42
Frankfurt - Hohenheim	0,00	0,00	0,00	0,67	0,17	0,27
Frankfurt - IIS Erlangen	1,00	0,00	0,00	1,00	0,26	0,41
Frankfurt - Muenster	0,00	0,00	0,00	1,00	0,11	0,19
Frankfurt - Potsdam	1,00	0,00	0,00	1,00	0,14	0,24
Frankfurt - TU Munich	0,66	1,00	0,79	0,77	1,00	0,87
Frankfurt - Wuerzburg	1,00	0,17	0,29	0,57	0,17	0,26
FU Berlin - Hohenheim	1,00	0,00	0,00	1,00	0,10	0,18
FU Berlin - IIS Erlangen	0,68	0,89	0,77	0,68	0,89	0,77
FU Berlin - Muenster	0,43	0,48	0,45	0,58	0,48	0,53
FU Berlin - Potsdam	0,73	0,96	0,83	0,73	0,96	0,83
FU Berlin - TU Munich	1,00	0,22	0,36	1,00	0,22	0,36
FU Berlin - Wuerzburg	0,61	0,61	0,61	0,81	0,81	0,81
Hohenheim - IIS Erlangen	0,52	0,25	0,34	0,43	0,17	0,25
Hohenheim - Muenster	0,61	0,31	0,41	0,63	0,42	0,50
Hohenheim - Potsdam	0,36	0,10	0,16	0,63	0,10	0,17
Hohenheim - TU Munich	0,00	0,00	0,00	0,45	0,13	0,20
Hohenheim - Wuerzburg	1,00	0,00	0,00	0,50	0,19	0,27
IIS Erlangen - Muenster	0,43	0,48	0,46	0,50	0,48	0,49
IIS Erlangen - Potsdam	0,76	0,77	0,76	0,78	0,86	0,82
IIS Erlangen - TU Munich	0,77	0,33	0,47	0,80	0,20	0,32
IIS Erlangen - Wuerzburg	0,60	0,56	0,58	0,78	0,66	0,71
Muenster - Potsdam	0,49	0,46	0,47	0,51	0,46	0,48
Muenster - TU Munich	0,75	0,26	0,39	0,82	0,21	0,33
Muenster - Wuerzburg	0,42	0,35	0,38	0,45	0,23	0,31
Potsdam - TU Munich	1,00	0,29	0,45	1,00	0,29	0,45
Potsdam - Wuerzburg	0,52	0,44	0,48	0,73	0,67	0,70
TU Munich - Wuerzburg	1,00	0,21	0,34	1,00	0,21	0,34
Macro Average	*0,68*	*0,34*	*0,37*	*0,77*	*0,38*	*0,44*
Standardabweichung	*0,30*	*0,31*	*0,26*	*0,20*	*0,30*	*0,23*
Micro Average	*0,62*	*0,39*	*0,48*	*0,70*	*0,43*	*0,53*

Tab. A.7: University Admission Matching-Ergebnisse Goldstandard Mehrheit

University Admission	**Triple-S**			**Triple-S2**		
	ProP	ProR	ProFW	ProP	ProR	ProFW
Cologne - Frankfurt	0,69	1,00	0,81	0,79	1,00	0,88
Cologne - FU Berlin	1,00	0,14	0,25	1,00	0,29	0,44
Cologne - Hohenheim	0,00	0,00	0,00	0,33	0,11	0,17
Cologne - IIS Erlangen	0,73	0,29	0,41	1,00	0,29	0,44
Cologne - Muenster	0,73	0,42	0,53	1,00	0,42	0,59
Cologne - Potsdam	1,00	0,35	0,52	1,00	0,35	0,52
Cologne - TU Munich	0,71	1,00	0,83	0,79	1,00	0,88
Cologne - Wuerzburg	1,00	0,18	0,31	0,57	0,18	0,28
Frankfurt - FU Berlin	1,00	0,44	0,62	1,00	0,44	0,62
Frankfurt - Hohenheim	0,00	0,00	0,00	0,45	0,18	0,26
Frankfurt - IIS Erlangen	1,00	0,00	0,00	1,00	0,38	0,55
Frankfurt - Muenster	0,00	0,00	0,00	1,00	0,24	0,38
Frankfurt - Potsdam	1,00	0,00	0,00	1,00	0,22	0,36
Frankfurt - TU Munich	0,54	1,00	0,70	0,63	1,00	0,77
Frankfurt - Wuerzburg	1,00	0,25	0,40	0,57	0,25	0,35
FU Berlin - Hohenheim	1,00	0,00	0,00	1,00	0,17	0,30
FU Berlin - IIS Erlangen	0,68	0,91	0,78	0,68	0,91	0,78
FU Berlin - Muenster	0,40	0,55	0,46	0,53	0,55	0,54
FU Berlin - Potsdam	0,73	1,00	0,84	0,73	1,00	0,84
FU Berlin - TU Munich	0,70	0,22	0,33	0,70	0,22	0,33
FU Berlin - Wuerzburg	0,55	0,72	0,62	0,59	0,88	0,71
Hohenheim - IIS Erlangen	0,52	0,36	0,43	0,32	0,19	0,24
Hohenheim - Muenster	0,53	0,35	0,43	0,48	0,46	0,47
Hohenheim - Potsdam	0,36	0,15	0,21	0,63	0,15	0,24
Hohenheim - TU Munich	0,00	0,00	0,00	0,31	0,13	0,19
Hohenheim - Wuerzburg	1,00	0,00	0,00	0,31	0,15	0,21
IIS Erlangen - Muenster	0,42	0,57	0,48	0,48	0,57	0,52
IIS Erlangen - Potsdam	0,76	0,87	0,81	0,73	0,93	0,82
IIS Erlangen - TU Munich	0,77	0,41	0,53	0,80	0,24	0,38
IIS Erlangen - Wuerzburg	0,45	0,54	0,49	0,60	0,64	0,62
Muenster - Potsdam	0,46	0,67	0,55	0,48	0,67	0,56
Muenster - TU Munich	0,64	0,40	0,49	0,67	0,30	0,41
Muenster - Wuerzburg	0,42	0,54	0,47	0,45	0,36	0,40
Potsdam - TU Munich	0,50	0,32	0,39	0,50	0,32	0,39
Potsdam - Wuerzburg	0,45	0,45	0,45	0,66	0,70	0,68
TU Munich - Wuerzburg	0,33	0,16	0,21	0,33	0,16	0,21
Macro Average	*0,61*	*0,40*	*0,40*	*0,67*	*0,45*	*0,48*
Standardabweichung	*0,30*	*0,32*	*0,27*	*0,23*	*0,29*	*0,21*
Micro Average	*0,58*	*0,50*	*0,54*	*0,63*	*0,54*	*0,58*

Tab. A.8: Birth Registration Matching-Ergebnisse Goldstandard 1

Birth Registration	**Triple-S**			**Triple-S2**		
	Precision	Recall	F-Wert	Precision	Recall	F-Wert
p31 - p32	1,00	0,17	0,29	1,00	0,17	0,29
p31 - p33	0,75	0,38	0,50	0,75	0,38	0,50
p31 - p34	0,43	0,30	0,35	0,50	0,30	0,38
p31 - p246	1,00	0,67	0,80	1,00	0,67	0,80
p31 - p247	0,50	0,20	0,29	1,00	0,20	0,00
p31 - p248	1,00	0,25	0,40	1,00	0,25	0,40
p31 - p249	1,00	0,00	0,00	1,00	0,00	0,00
p31 - p250	1,00	0,40	0,57	1,00	0,40	0,57
p32 - p33	0,43	0,60	0,50	0,38	0,60	0,46
p32 - p34	0,50	0,14	0,22	0,50	0,14	0,22
p32 - p246	1,00	0,60	0,75	1,00	0,40	0,57
p32 - p247	1,00	0,33	0,50	1,00	0,33	0,50
p32 - p248	1,00	0,20	0,33	1,00	0,20	0,33
p32 - p249	1,00	0,00	0,00	1,00	0,00	0,00
p32 - p250	1,00	0,14	0,25	1,00	0,14	0,25
p33 - p34	0,75	0,50	0,60	1,00	0,50	0,67
p33 - p246	1,00	0,67	0,80	1,00	0,67	0,80
p33 - p247	1,00	0,25	0,40	1,00	0,25	0,40
p33 - p248	1,00	0,25	0,40	1,00	0,25	0,40
p33 - p249	1,00	0,00	0,00	1,00	0,00	0,00
p33 - p250	1,00	0,20	0,33	1,00	0,20	0,33
p34 - p246	0,67	0,50	0,57	1,00	0,50	0,67
p34 - p247	1,00	0,00	0,00	1,00	0,00	0,00
p34 - p248	1,00	0,00	0,00	1,00	0,00	0,00
p34 - p249	0,00	0,00	0,00	0,00	0,00	0,00
p34 - p250	0,50	0,25	0,33	1,00	0,25	0,40
p246 - p247	0,65	0,87	0,74	0,81	0,87	0,84
p246 - p248	0,70	0,93	0,80	0,72	0,87	0,79
p246 - p249	0,77	0,71	0,74	0,83	0,71	0,77
p246 - p250	0,44	0,58	0,50	0,57	0,67	0,62
p247 - p248	0,81	0,88	0,84	0,87	0,83	0,85
p247 - p249	0,79	0,85	0,81	0,83	0,77	0,80
p247 - p250	0,63	0,56	0,59	0,56	0,56	0,56
p248 - p249	0,75	0,92	0,83	0,76	1,00	0,87
p248 - p250	0,67	0,67	0,67	0,56	0,56	0,56
p249 - p250	0,50	0,27	0,35	0,57	0,36	0,44
Macro Average	*0,78*	*0,40*	*0,45*	*0,84*	*0,39*	*0,45*
Standardabweichung	*0,25*	*0,29*	*0,27*	*0,24*	*0,28*	*0,28*
Micro Average	*0,69*	*0,52*	*0,60*	*0,74*	*0,52*	*0,61*

Tab. A.9: Birth Registration Matching-Ergebnisse Goldstandard 2

Birth Registration	Triple-S			Triple-S2		
	Precision	Recall	F-Wert	Precision	Recall	F-Wert
p31 - p32	1,00	0,06	0,12	1,00	0,06	0,12
p31 - p33	0,50	0,20	0,29	0,50	0,20	0,29
p31 - p34	0,86	0,23	0,36	1,00	0,23	0,38
p31 - p246	1,00	0,09	0,16	1,00	0,09	0,16
p31 - p247	0,50	0,04	0,07	1,00	0,04	0,07
p31 - p248	1,00	0,04	0,07	1,00	0,04	0,07
p31 - p249	1,00	0,00	0,00	1,00	0,00	0,00
p31 - p250	0,50	0,05	0,09	0,50	0,05	0,09
p32 - p33	0,43	0,23	0,30	0,38	0,23	0,29
p32 - p34	0,50	0,07	0,13	0,50	0,07	0,13
p32 - p246	1,00	0,19	0,32	1,00	0,13	0,22
p32 - p247	1,00	0,08	0,15	1,00	0,08	0,15
p32 - p248	1,00	0,04	0,08	1,00	0,04	0,08
p32 - p249	1,00	0,00	0,00	1,00	0,00	0,00
p32 - p250	1,00	0,07	0,13	1,00	0,07	0,13
p33 - p34	0,75	0,38	0,50	1,00	0,38	0,55
p33 - p246	1,00	0,18	0,31	1,00	0,18	0,31
p33 - p247	1,00	0,08	0,14	1,00	0,08	0,14
p33 - p248	1,00	0,08	0,15	1,00	0,08	0,15
p33 - p249	1,00	0,00	0,00	1,00	0,00	0,00
p33 - p250	1,00	0,08	0,15	1,00	0,08	0,15
p34 - p246	0,67	0,33	0,44	1,00	0,33	0,50
p34 - p247	1,00	0,00	0,00	1,00	0,00	0,00
p34 - p248	1,00	0,00	0,00	1,00	0,00	0,00
p34 - p249	0,00	0,00	0,00	0,00	0,00	0,00
p34 - p250	0,50	0,14	0,22	1,00	0,14	0,25
p246 - p247	0,70	0,64	0,67	0,81	0,59	0,68
p246 - p248	0,70	0,64	0,67	0,72	0,59	0,65
p246 - p249	0,69	0,50	0,58	0,75	0,50	0,60
p246 - p250	0,44	0,44	0,44	0,50	0,44	0,47
p247 - p248	0,77	0,95	0,85	0,83	0,90	0,86
p247 - p249	0,86	0,67	0,75	0,92	0,61	0,73
p247 - p250	0,75	0,32	0,44	0,78	0,37	0,50
p248 - p249	0,75	0,67	0,71	0,76	0,72	0,74
p248 - p250	0,78	0,37	0,50	0,78	0,37	0,50
p249 - p250	0,67	0,25	0,36	0,71	0,31	0,43
Macro Average	*0,79*	*0,22*	*0,28*	*0,85*	*0,22*	*0,29*
Standardabweichung	*0,24*	*0,24*	*0,24*	*0,23*	*0,23*	*0,25*
Micro Average	*0,71*	*0,24*	*0,36*	*0,76*	*0,24*	*0,36*

Tab. A.10: Birth Registration Matching-Ergebnisse Goldstandard 3

Birth Registration	Triple-S			Triple-S2		
	Precision	Recall	F-Wert	Precision	Recall	F-Wert
p31 - p32	1,00	0,07	0,13	1,00	0,07	0,13
p31 - p33	0,25	0,08	0,13	0,25	0,08	0,13
p31 - p34	0,14	0,17	0,15	0,17	0,17	0,17
p31 - p246	0,50	0,33	0,40	0,50	0,33	0,40
p31 - p247	0,00	0,00	0,00	0,00	0,00	0,00
p31 - p248	1,00	0,33	0,50	1,00	0,33	0,50
p31 - p249	1,00	1,00	1,00	1,00	1,00	1,00
p31 - p250	1,00	0,67	0,80	1,00	0,67	0,80
p32 - p33	0,43	0,50	0,46	0,38	0,50	0,43
p32 - p34	0,50	0,33	0,40	0,50	0,33	0,40
p32 - p246	0,67	1,00	0,80	1,00	1,00	1,00
p32 - p247	1,00	0,67	0,80	1,00	0,67	0,80
p32 - p248	1,00	0,33	0,50	1,00	0,33	0,50
p32 - p249	1,00	0,00	0,00	1,00	0,00	0,00
p32 - p250	1,00	0,25	0,40	1,00	0,25	0,40
p33 - p34	0,75	0,43	0,55	1,00	0,43	0,60
p33 - p246	0,50	1,00	0,67	0,50	1,00	0,67
p33 - p247	1,00	1,00	1,00	1,00	1,00	1,00
p33 - p248	1,00	1,00	1,00	1,00	1,00	1,00
p33 - p249	1,00	0,00	0,00	1,00	0,00	0,00
p33 - p250	1,00	1,00	1,00	1,00	1,00	1,00
p34 - p246	0,67	0,67	0,67	1,00	0,67	0,80
p34 - p247	1,00	0,00	0,00	1,00	0,00	0,00
p34 - p248	1,00	0,00	0,00	1,00	0,00	0,00
p34 - p249	0,00	0,00	0,00	0,00	0,00	0,00
p34 - p250	0,50	0,25	0,33	1,00	0,25	0,40
p246 - p247	0,60	0,86	0,71	0,75	0,86	0,80
p246 - p248	0,70	0,88	0,78	0,72	0,81	0,76
p246 - p249	0,62	0,67	0,64	0,67	0,67	0,67
p246 - p250	0,44	0,70	0,54	0,57	0,80	0,67
p247 - p248	0,81	0,84	0,82	0,87	0,80	0,83
p247 - p249	0,50	0,78	0,61	0,58	0,78	0,67
p247 - p250	0,38	0,50	0,43	0,33	0,50	0,40
p248 - p249	0,75	0,92	0,83	0,76	1,00	0,87
p248 - p250	0,78	0,78	0,78	0,67	0,67	0,67
p249 - p250	0,67	0,57	0,62	0,71	0,71	0,71
Macro Average	*0,70*	*0,52*	*0,51*	*0,75*	*0,52*	*0,53*
Standardabweichung	*0,30*	*0,36*	*0,31*	*0,31*	*0,36*	*0,33*
Micro Average	*0,62*	*0,56*	*0,59*	*0,67*	*0,56*	*0,61*

Tab. A.11: Birth Registration Matching-Ergebnisse Goldstandard 4

Birth Registration	Triple-S			Triple-S2		
	Precision	Recall	F-Wert	Precision	Recall	F-Wert
p31 - p32	1,00	0,08	0,14	1,00	0,08	0,14
p31 - p33	0,25	0,14	0,18	0,25	0,14	0,18
p31 - p34	0,14	0,33	0,20	0,17	0,33	0,22
p31 - p246	0,50	0,50	0,50	0,50	0,50	0,50
p31 - p247	0,50	0,17	0,25	1,00	0,17	0,00
p31 - p248	1,00	0,20	0,33	1,00	0,20	0,33
p31 - p249	1,00	0,00	0,00	1,00	0,00	0,00
p31 - p250	0,50	0,14	0,22	0,50	0,14	0,22
p32 - p33	0,14	0,25	0,18	0,13	0,25	0,17
p32 - p34	0,50	0,07	0,13	0,50	0,07	0,13
p32 - p246	0,67	0,50	0,57	1,00	0,50	0,67
p32 - p247	1,00	0,29	0,44	1,00	0,29	0,44
p32 - p248	1,00	0,14	0,25	1,00	0,14	0,25
p32 - p249	1,00	0,00	0,00	1,00	0,00	0,00
p32 - p250	1,00	0,13	0,22	1,00	0,13	0,22
p33 - p34	0,50	0,50	0,50	0,67	0,50	0,57
p33 - p246	0,00	0,00	0,00	0,00	0,00	0,00
p33 - p247	0,00	0,00	0,00	0,00	0,00	0,00
p33 - p248	0,00	0,00	0,00	0,00	0,00	0,00
p33 - p249	1,00	0,00	0,00	1,00	0,00	0,00
p33 - p250	0,00	0,00	0,00	0,00	0,00	0,00
p34 - p246	0,33	1,00	0,50	0,50	1,00	0,67
p34 - p247	1,00	0,00	0,00	1,00	0,00	0,00
p34 - p248	1,00	0,00	0,00	1,00	0,00	0,00
p34 - p249	0,00	0,00	0,00	0,00	0,00	0,00
p34 - p250	0,50	0,20	0,29	1,00	0,20	0,33
p246 - p247	0,65	0,93	0,76	0,81	0,93	0,87
p246 - p248	0,75	0,88	0,81	0,78	0,82	0,80
p246 - p249	0,77	0,59	0,67	0,83	0,59	0,69
p246 - p250	0,44	0,50	0,47	0,57	0,57	0,57
p247 - p248	0,77	0,80	0,78	0,83	0,76	0,79
p247 - p249	0,86	0,80	0,83	0,92	0,73	0,81
p247 - p250	0,63	0,45	0,53	0,56	0,45	0,50
p248 - p249	0,81	0,76	0,79	0,82	0,82	0,82
p248 - p250	0,56	0,50	0,53	0,56	0,50	0,53
p249 - p250	0,50	0,25	0,33	0,57	0,33	0,42
Macro Average	*0,59*	*0,31*	*0,32*	*0,65*	*0,31*	*0,33*
Standardabweichung	*0,35*	*0,31*	*0,27*	*0,36*	*0,30*	*0,30*
Micro Average	*0,62*	*0,44*	*0,51*	*0,67*	*0,44*	*0,53*

Tab. A.12: Birth Registration Matching-Ergebnisse Goldstandard 5

Birth Registration	**Triple-S**			**Triple-S2**		
	Precision	Recall	F-Wert	Precision	Recall	F-Wert
p31 - p32	1,00	0,50	0,67	1,00	0,50	0,67
p31 - p33	0,50	1,00	0,67	0,50	1,00	0,67
p31 - p34	0,14	1,00	0,25	0,17	1,00	0,29
p31 - p246	0,50	1,00	0,67	0,50	1,00	0,67
p31 - p247	0,50	0,50	0,50	1,00	0,50	0,67
p31 - p248	1,00	1,00	1,00	1,00	1,00	1,00
p31 - p249	1,00	1,00	1,00	1,00	1,00	1,00
p31 - p250	0,50	1,00	0,67	0,50	1,00	0,67
p32 - p33	0,29	1,00	0,44	0,25	1,00	0,40
p32 - p34	0,50	1,00	0,67	0,50	1,00	0,67
p32 - p246	0,67	1,00	0,80	1,00	1,00	1,00
p32 - p247	1,00	1,00	1,00	1,00	1,00	1,00
p32 - p248	1,00	0,50	0,67	1,00	0,50	0,67
p32 - p249	1,00	0,00	0,00	1,00	0,00	0,00
p32 - p250	1,00	0,50	0,67	1,00	0,50	0,67
p33 - p34	0,25	1,00	0,40	0,33	1,00	0,50
p33 - p246	0,50	1,00	0,67	0,50	1,00	0,67
p33 - p247	1,00	1,00	1,00	1,00	1,00	1,00
p33 - p248	1,00	1,00	1,00	1,00	1,00	1,00
p33 - p249	1,00	1,00	1,00	1,00	1,00	1,00
p33 - p250	1,00	1,00	1,00	1,00	1,00	1,00
p34 - p246	0,33	1,00	0,50	0,50	1,00	0,67
p34 - p247	1,00	0,00	0,00	1,00	0,00	0,00
p34 - p248	1,00	0,00	0,00	1,00	0,00	0,00
p34 - p249	0,00	1,00	0,00	0,00	1,00	0,00
p34 - p250	0,50	1,00	0,67	1,00	1,00	1,00
p246 - p247	0,60	1,00	0,75	0,75	1,00	0,86
p246 - p248	0,55	0,92	0,69	0,61	0,92	0,73
p246 - p249	0,77	0,71	0,74	0,83	0,71	0,77
p246 - p250	0,44	0,64	0,52	0,57	0,73	0,64
p247 - p248	0,69	0,86	0,77	0,78	0,86	0,82
p247 - p249	0,71	0,83	0,77	0,83	0,83	0,83
p247 - p250	0,63	0,56	0,59	0,56	0,56	0,56
p248 - p249	0,69	0,92	0,79	0,71	1,00	0,83
p248 - p250	0,56	0,63	0,59	0,56	0,63	0,59
p249 - p250	0,67	0,50	0,57	0,71	0,63	0,67
Macro Average	*0,68*	*0,79*	*0,63*	*0,74*	*0,80*	*0,67*
Standardabweichung	*0,28*	*0,30*	*0,29*	*0,28*	*0,30*	*0,30*
Micro Average	*0,59*	*0,78*	*0,67*	*0,66*	*0,80*	*0,72*

Tab. A.13: Birth Registration Matching-Ergebnisse Goldstandard Gesamt

Birth Registration	Triple-S			Triple-S2		
	ProP	ProR	ProFW	ProP	ProR	ProFW
p31 - p32	1,00	0,10	0,18	1,00	0,10	0,18
p31 - p33	0,75	0,23	0,35	0,75	0,23	0,35
p31 - p34	0,80	0,26	0,39	1,00	0,26	0,41
p31 - p246	1,00	0,22	0,36	1,00	0,22	0,36
p31 - p247	0,57	0,09	0,16	1,00	0,09	0,17
p31 - p248	1,00	0,13	0,23	1,00	0,13	0,23
p31 - p249	1,00	0,00	0,00	1,00	0,00	0,00
p31 - p250	1,00	0,19	0,32	1,00	0,19	0,32
p32 - p33	0,57	0,40	0,47	0,50	0,40	0,44
p32 - p34	0,63	0,13	0,21	0,63	0,13	0,21
p32 - p246	1,00	0,41	0,59	1,00	0,34	0,51
p32 - p247	1,00	0,23	0,38	1,00	0,23	0,38
p32 - p248	1,00	0,13	0,22	1,00	0,13	0,22
p32 - p249	1,00	0,00	0,00	1,00	0,00	0,00
p32 - p250	1,00	0,14	0,25	1,00	0,14	0,25
p33 - p34	0,80	0,46	0,59	0,80	0,46	0,59
p33 - p246	1,00	0,35	0,52	1,00	0,35	0,52
p33 - p247	1,00	0,19	0,32	1,00	0,19	0,32
p33 - p248	1,00	0,21	0,35	1,00	0,21	0,35
p33 - p249	1,00	0,00	0,00	1,00	0,00	0,00
p33 - p250	1,00	0,19	0,32	1,00	0,19	0,32
p34 - p246	0,73	0,53	0,62	1,00	0,53	0,70
p34 - p247	1,00	0,00	0,00	1,00	0,00	0,00
p34 - p248	1,00	0,00	0,00	1,00	0,00	0,00
p34 - p249	0,00	0,00	0,00	0,00	0,00	0,00
p34 - p250	0,63	0,24	0,34	1,00	0,24	0,38
p246 - p247	0,81	0,83	0,82	0,91	0,82	0,86
p246 - p248	0,82	0,83	0,82	0,84	0,78	0,81
p246 - p249	0,84	0,63	0,72	0,89	0,63	0,73
p246 - p250	0,56	0,56	0,56	0,69	0,63	0,66
p247 - p248	0,87	0,86	0,87	0,91	0,83	0,87
p247 - p249	0,90	0,78	0,83	0,94	0,73	0,82
p247 - p250	0,80	0,44	0,57	0,81	0,46	0,59
p248 - p249	0,91	0,83	0,87	0,92	0,90	0,91
p248 - p250	0,91	0,55	0,68	0,90	0,51	0,65
p249 - p250	0,75	0,33	0,46	0,79	0,43	0,55
Macro Average	*0,85*	*0,32*	*0,40*	*0,90*	*0,32*	*0,41*
Standardabweichung	*0,20*	*0,27*	*0,27*	*0,19*	*0,26*	*0,27*
Micro Average	*0,82*	*0,43*	*0,56*	*0,87*	*0,43*	*0,57*

Tab. A.14: Birth Registration Matching-Ergebnisse Goldstandard Mehrheit

Birth Registration	**Triple-S**			**Triple-S2**		
	ProP	ProR	ProFW	ProP	ProR	ProFW
p31 - p32	1,00	0,15	0,26	1,00	0,15	0,26
p31 - p33	0,54	0,27	0,36	0,54	0,27	0,36
p31 - p34	0,22	0,33	0,26	0,25	0,33	0,29
p31 - p246	0,63	0,63	0,63	0,63	0,63	0,63
p31 - p247	0,57	0,40	0,47	1,00	0,40	0,57
p31 - p248	1,00	0,63	0,77	1,00	0,63	0,77
p31 - p249	1,00	1,00	1,00	1,00	1,00	1,00
p31 - p250	0,45	1,00	0,63	0,63	1,00	0,77
p32 - p33	0,50	0,45	0,47	0,43	0,45	0,44
p32 - p34	0,63	0,26	0,37	0,63	0,26	0,37
p32 - p246	1,00	1,00	1,00	1,00	1,00	1,00
p32 - p247	1,00	1,00	1,00	1,00	1,00	1,00
p32 - p248	1,00	0,50	0,67	1,00	0,50	0,67
p32 - p249	1,00	0,00	0,00	1,00	0,00	0,00
p32 - p250	1,00	0,38	0,56	1,00	0,38	0,56
p33 - p34	0,80	0,63	0,71	0,80	0,63	0,71
p33 - p246	0,57	1,00	0,73	0,57	1,00	0,73
p33 - p247	1,00	1,00	1,00	1,00	1,00	1,00
p33 - p248	1,00	1,00	1,00	1,00	1,00	1,00
p33 - p249	1,00	1,00	1,00	1,00	1,00	1,00
p33 - p250	1,00	1,00	1,00	1,00	1,00	1,00
p34 - p246	0,73	1,00	0,84	1,00	1,00	1,00
p34 - p247	1,00	0,00	0,00	1,00	0,00	0,00
p34 - p248	1,00	0,00	0,00	1,00	0,00	0,00
p34 - p249	0,00	0,00	0,00	0,00	0,00	0,00
p34 - p250	0,63	0,45	0,53	1,00	0,45	0,63
p246 - p247	0,81	0,94	0,87	0,91	0,94	0,93
p246 - p248	0,81	0,93	0,87	0,81	0,87	0,84
p246 - p249	0,84	0,69	0,76	0,89	0,69	0,78
p246 - p250	0,56	0,66	0,61	0,69	0,75	0,72
p247 - p248	0,87	0,87	0,87	0,91	0,83	0,87
p247 - p249	0,90	0,85	0,87	0,94	0,80	0,87
p247 - p250	0,79	0,58	0,67	0,79	0,58	0,67
p248 - p249	0,87	0,92	0,89	0,88	1,00	0,93
p248 - p250	0,76	0,68	0,72	0,68	0,61	0,64
p249 - p250	0,75	0,43	0,55	0,79	0,55	0,65
Macro Average	*0,78*	*0,63*	*0,64*	*0,82*	*0,63*	*0,65*
Standardabweichung	*0,24*	*0,34*	*0,31*	*0,24*	*0,34*	*0,31*
Micro Average	*0,77*	*0,69*	*0,73*	*0,80*	*0,69*	*0,74*

A.2 Anhang zur ähnlichkeitsbasierten Suche

A.2.1 Berechnungen für das LS3-QueryAll Beispiel

Die folgenden Matrizen zeigen die vollständige Beispielrechnung des LS3-Query All Algorithmus aus Kapitel 5.3.1. Abbildung A.1 zeigt zunächst die Term-Dokument Matrix sowie die gewichtete Term-Dokument Matrix, die sich aus den drei Modellen in Abbildung 5.2 ergeben. D_1 referenziert das Dienstreisemodell, während D_2 und D_3 auf die beiden Warenversandmodelle verweisen. In den Abbildungen A.2 und A.3 sind daraufhin die Singulärwertzerlegung und die auf $k = 2$ reduzierte Singulärwertzerlegung dargestellt. Schließlich zeigt Abbildung A.4 die LSSM-Matrix mit den Ähnlichkeitswerten der drei Modelle. Wird etwa $\theta = 0,8$ gewählt, so ergibt sich für D_1 als ähnliches Modell D_1, für D_2 ergeben sich die Modelle D_2 und D_3 als ähnlich und entsprechend für D_3 ebenfalls D_2 und D_3.

	D_1	D_2	D_3
return	1,0	0,0	0,0
applic	5,0	0,0	0,0
reject	1,0	0,0	0,0
fill	1,0	0,0	0,0
vacat	1,0	0,0	0,0
submit	1,0	0,0	0,0
approv	1,0	0,0	0,0
order	0,0	4,0	3,0
receiv	0,0	2,0	2,0
pack	0,0	1,0	0,0
good	0,0	2,0	1,0
ship	0,0	1,0	1,0
close	0,0	1,0	1,0
monei	0,0	1,0	0,0
gener	0,0	1,0	0,0
prepar	0,0	1,0	0,0
send	0,0	1,0	1,0
invoic	0,0	1,0	1,0
purchas	0,0	0,0	2,0
creat	0,0	0,0	1,0
payment	0,0	0,0	1,0

(a) Term-Dokument Matrix

	D_1	D_2	D_3
return	1,00	0,00	0,00
applic	2,58	0,00	0,00
reject	1,00	0,00	0,00
fill	1,00	0,00	0,00
vacat	1,00	0,00	0,00
submit	1,00	0,00	0,00
approv	1,00	0,00	0,00
order	0,00	0,97	0,68
receiv	0,00	0,58	0,58
pack	0,00	1,00	0,00
good	0,00	0,81	0,33
ship	0,00	0,37	0,37
close	0,00	0,37	0,37
monei	0,00	1,00	0,00
gener	0,00	1,00	0,00
prepar	0,00	1,00	0,00
send	0,00	0,37	0,37
invoic	0,00	0,37	0,37
purchas	0,00	0,00	1,58
creat	0,00	0,00	1,00
payment	0,00	0,00	1,00

(b) Gewichtete Term-Dokument Matrix

Abb. A.1: Term-Dokument Matrizen zur LS3-QueryAll-Beispielrechnung

$$\begin{pmatrix} -0,28 & 0,00 & 0,00 \\ -0,73 & 0,00 & 0,00 \\ -0,28 & 0,00 & 0,00 \\ -0,28 & 0,00 & 0,00 \\ -0,28 & 0,00 & 0,00 \\ -0,28 & 0,00 & 0,00 \\ -0,28 & 0,00 & 0,00 \\ 0,00 & 0,41 & -0,06 \\ 0,00 & 0,29 & 0,03 \\ 0,00 & 0,27 & -0,31 \\ 0,00 & 0,29 & -0,13 \\ 0,00 & 0,18 & 0,02 \\ 0,00 & 0,18 & 0,02 \\ 0,00 & 0,27 & -0,31 \\ 0,00 & 0,27 & -0,31 \\ 0,00 & 0,27 & -0,31 \\ 0,00 & 0,18 & 0,02 \\ 0,00 & 0,18 & 0,02 \\ 0,00 & 0,37 & 0,57 \\ 0,00 & 0,23 & 0,36 \\ 0,00 & 0,23 & 0,36 \end{pmatrix}$$

(a) Term-Matrix T

$$\begin{pmatrix} 3,56 & 0,00 & 0,00 \\ 0,00 & 2,84 & 0,00 \\ 0,00 & 0,00 & 2,10 \end{pmatrix}$$

(b) Singulär-Matrix Σ

$$\begin{pmatrix} -1,00 & 0,00 & 0,00 \\ 0,00 & 0,75 & 0,66 \\ 0,00 & -0,66 & 0,75 \end{pmatrix}$$

(c) Dokument-Matrix D^T

Abb. A.2: Singulärwertzerlegung zur LS3-QueryAll-Beispielrechnung

$$\begin{pmatrix} -0,28 & 4,90E-17 \\ -0,73 & -3,97E-17 \\ -0,28 & 1,07E-17 \\ -0,28 & 1,07E-17 \\ -0,28 & 1,07E-17 \\ -0,28 & 1,07E-17 \\ -0,28 & 1,07E-17 \\ 0,00 & 4,15E-01 \\ 0,00 & 2,91E-01 \\ 0,00 & 2,66E-01 \\ 0,00 & 2,91E-01 \\ 0,00 & 1,83E-01 \\ 0,00 & 1,83E-01 \\ 0,00 & 2,66E-01 \\ 0,00 & 2,66E-01 \\ 0,00 & 2,66E-01 \\ 0,00 & 1,83E-01 \\ 0,00 & 1,83E-01 \\ 0,00 & 3,67E-01 \\ 0,00 & 2,31E-01 \\ 0,00 & 2,31E-01 \end{pmatrix}$$

(a) Term-Matrix T_k

$$\begin{pmatrix} 3,56 & 0,00 \\ 0,00 & 2,84 \end{pmatrix}$$

(b) Singulär-Matrix Σ_k

$$\begin{pmatrix} -1,00 & 0,00 & 0,00 \\ 0,00 & 0,75 & 0,66 \end{pmatrix}$$

(c) Dokument-Matrix D^T

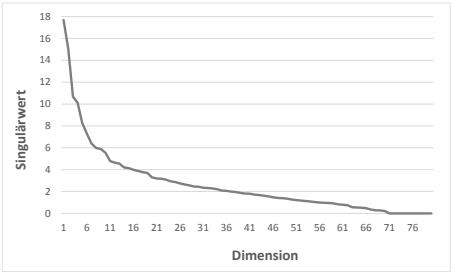
Abb. A.3: Reduzierte Singulärwertzerlegung zur LS3-QueryAll-Beispielrechnung ($k = 2$)

$$\begin{array}{c} \\ D_1 \\ D_2 \\ D_3 \end{array} \begin{array}{c} \begin{array}{ccc} D_1 & D_2 & D_3 \end{array} \\ \begin{pmatrix} 1,0 & 0,5 & 0,5 \\ 0,5 & 1,0 & 1,0 \\ 0,5 & 1,0 & 1,0 \end{pmatrix} \end{array}$$

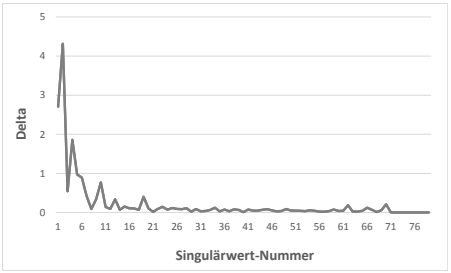
Abb. A.4: LSSM-Matrix zur LS3-QueryAll-Beispielrechnung

A.2.2 Singulärwerte der LS3 Evaluationsdatensätze

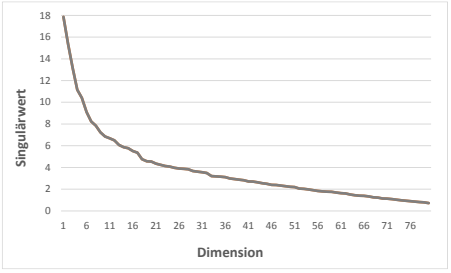
Auf der folgenden Seite sind in Abbildung A.5 die Singulärwerte sowie die Differenz zwischen zwei aufeinander folgenden Singulärwerten für die Evaluation der in Kapitel 5.3 beschriebenen LS3-Verfahren abgebildet.



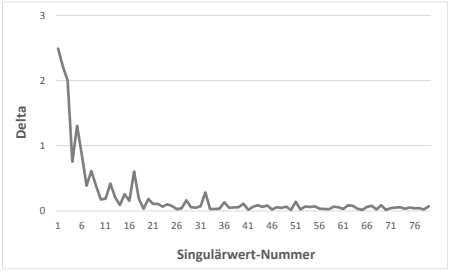
(a) Singulärwerte des ersten Datensatzes



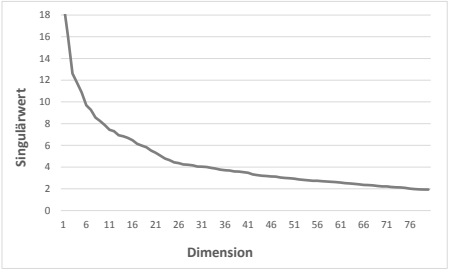
(b) Differenz der Singulärwerte des ersten Datensatzes



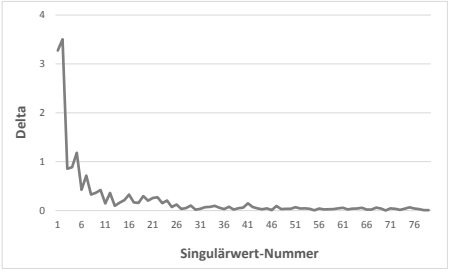
(c) Singulärwerte des zweiten Datensatzes



(d) Differenz der Singulärwerte des zweiten Datensatzes



(e) Singulärwerte des dritten Datensatzes



(f) Differenz der Singulärwerte des dritten Datensatzes

Abb. A.5: Singulärwerte und Differenz von zwei aufeinander folgenden Singulärwerten für die Evaluationsdatensätze der LS3-Verfahren

Literaturverzeichnis

Abbas, S. und H. Seba (2012). „A Module-based Approach for Structural Matching of Process Models“. In: *5th IEEE International Conference on Service-Oriented Computing and Applications (SOCA), Taipei, Taiwan*. IEEE Computer Society, S. 1–8. DOI: 10.1109/SOCA.2012.6449441.

Akkiraju, R. und A. Ivan (2010). „Discovering Business Process Similarities: An Empirical Study with SAP Best Practice Business Processes.“ In: *8th International Conference on Service Oriented Computing (ICSOC), San Francisco, USA*. Hrsg. von P. P. Maglio, M. Weske, J. Yang und M. Fantinato. Bd. 6470. Lecture Notes in Computer Science. Springer, S. 515–526. DOI: 10.1007/978-3-642-17358-5_35.

Antunes, G., M. Bakhshandeh, J. Borbinha, J. Cardoso, S. Dadashnia, C. D. Francescomarino, M. Dragoni, P. Fettke, A. Gal, C. Ghidini, P. Hake, A. Khiat, C. Klinkmüller, E. Kuss, H. Leopold, P. Loos, C. Meilicke, T. Niesen, C. Pesquita, T. Péus, A. Schoknecht, E. Sheetrit, A. Sonntag, H. Stuckenschmidt, T. Thaler, I. Weber und M. Weidlich (2015). „The Process Model Matching Contest 2015“. In: *6th International Workshop on Enterprise Modelling and Information Systems Architectures (EMISA 2015), Innsbruck, Österreich*. Hrsg. von J. Kolb, H. Leopold und J. Mendling. Bd. P-248. Lecture Notes in Informatics. Gesellschaft für Informatik, S. 127–155.

Awad, A. (2007). „BPMN-Q: A Language to Query Business Processes“. In: *2nd International Workshop on Enterprise Modelling and Information Systems Architectures - Concepts and Applications (EMISA), St. Goar, Deutschland*. Hrsg. von M. Reichert, S. Strecker und K. Turowski. Bd. P-119. Lecture Notes in Informatics. Gesellschaft für Informatik, S. 115–128.

Awad, A., A. Polyvyanyy und M. Weske (2008). „Semantic Querying of Business Process Models“. In: *12th International IEEE Enterprise Distributed Object Computing Conference (EDOC), München, Deutschland*. Hrsg. von C. Chi, D. Gasevic und W. van den Heuvel. IEEE Computer Society, S. 85–94. DOI: 10.1109/EDOC.2008.11.

Awad, A. und S. Sakr (2010). „Querying Graph-Based Repositories of Business Process Models“. In: *15th International Conference on Database Systems for Advanced Applications (DASFAA), Tsukuba, Japan*. Hrsg. von M. Yoshikawa, X. Meng, T. Yumoto, Q. Ma, L. Sun und C. Watanabe. Bd. 6193. Lecture Notes in Computer Science. Springer, S. 33–44. DOI: 10.1007/978-3-642-14589-6_4.

Bae, J., J. Caverlee, L. Liu und H. Yan (2006a). „Process Mining by Measuring Process Block Similarity“. In: *Business Process Management Workshops, Wien, Österreich*. Hrsg. von J. Eder und S. Dustdar. Bd. 4103. Lecture Notes in Computer Science. Springer, S. 141–152. DOI: 10.1007/11837862_15.

Bae, J., L. Liu, J. Caverlee und W. B. Rouse (2006b). „Process Mining, Discovery, and Integration using Distance Measures“. In: *International Conference on Web Services (ICWS), Chicago, USA*. IEEE Computer Society, S. 479–488. DOI: 10.1109/ICWS.2006.105.

Bae, J., L. Liu, J. Caverlee, L.-J. Zhang und H. Bae (2007). „Development of Distance Measures for Process Mining, Discovery and Integration“. In: *International Journal of Web Service Research* 4.4, S. 1–17. DOI: 10.4018/jwsr.2007100101.

Baroni, M., G. Dinu und G. Kruszewski (2014). „Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors“. In: *52nd Annual Meeting of the Association for Computational Linguistics (ACL), Baltimore, USA*. Hrsg. von A. Fraser und Y. Liu. The Association for Computer Linguistics, S. 238–247. DOI: 10.3115/v1/p14-1023.

Baumann, M. H., M. Baumann, S. Schönig und S. Jablonski (2014). „Towards Multi-perspective Process Model Similarity Matching“. In: *10th International Workshop on Enterprise and Organizational Modeling and Simulation (EOMAS), Thessaloniki, Griechenland*. Hrsg. von J. Barjis und R. Pergl. Bd. 191. Lecture Notes in Business Information Processing. Springer, S. 21–37. DOI: 10.1007/978-3-662-44860-1_2.

Baumann, M., M. H. Baumann und S. Jablonski (2015a). „On Behavioral Process Model Similarity Matching: A Centroid-based Approach“. In: *10th International Multi-Conference on Computing in the Global Information Technology (ICCGI), St. Julians, Malta*. Hrsg. von H. Kaindl, G. Kálmán und D. Tamir. IARIA XPS Press.

Baumann, M., M. H. Baumann, S. Schönig und S. Jablonski (2015b). „Resource-Aware Process Model Similarity Matching“. In: *Service-Oriented Computing Workshops (ICSOC), Paris, Frankreich*. Hrsg. von F. Toumani, B. Pernici, D. Grigori, D. Benslimane, J. Mendling, N. Ben Hadj-Alouane, B. Blake, O. Perrin, I. Saleh Moustafa und S. Bhiri. Bd. 8954. Lecture Notes in Computer Science. Springer, S. 96–107. DOI: 10.1007/978-3-319-22885-3_9.

Becker, J., P. Bergener, D. Breuker und M. Räckers (2011). „On Measures of Behavioral Distance between Business Processes“. In: *10. Internationale Tagung Wirtschaftsinformatik (WI), Zürich, Schweiz*. Hrsg. von A. Bernstein und G. Schwabe. Lulu.com, S. 665–674.

Becker, J., D. Breuker, P. Delfmann, H. Dietrich und S. Matthias (2012). „Identifying Business Process Activity Mappings by Optimizing Behavioral Similarity“. In: *18th Americas Conference on Information Systems (AMCIS), Seattle, USA*. Association for Information Systems.

Becker, J., M. Rosemann und C. von Uthmann (2000). „Guidelines of Business Process Modeling“. In: *Business Process Management, Models, Techniques, and Empirical Studies*. Hrsg. von W. M. P. van der Aalst, J. Desel und A. Oberweis. Bd. 1806. Lecture Notes in Computer Science. Springer, S. 30–49. DOI: 10.1007/3-540-45594-9_3.

Becker, J., B. Weiß und A. Winkelmann (2010). „Utility vs. Efforts of Business Process Modeling — An Exploratory Survey in the Financial Sector“. In: *Multikonferenz Wirtschaftsinformatik (MKWI), Göttingen, Deutschland*. Hrsg. von M. Schumann, L. M. Kolbe, M. H. Breitner und A. Frerichs. Universitätsverlag Göttingen, S. 41–54.

Becker, M. und R. Laue (2012). „A comparative survey of business process similarity measures“. In: *Computers in Industry* 63.2, S. 148–167. DOI: 10.1016/j.compind.2011.11.003.

Beeri, C., A. Eyal, S. Kamenkovich und T. Milo (2006). „Querying Business Processes“. In: *32nd International Conference on Very Large Data Bases, Seoul, Korea*. Hrsg. von U. Dayal, K. Whang, D. B. Lomet, G. Alonso, G. M. Lohman, M. L. Kersten, S. K. Cha und Y. Kim. ACM, S. 343–354.

Beeri, C., A. Eyal, S. Kamenkovich und T. Milo (2008). „Querying business processes with BP-QL“. In: *Information Systems* 33.6, S. 477–507. DOI: 10.1016/j.is.2008.02.005.

Beheshti, S., B. Benatallah, H. R. M. Nezhad und S. Sakr (2011). „A Query Language for Analyzing Business Processes Execution“. In: *9th International Conference on Business Process Management (BPM), Clermont-Ferrand, Frankreich*. Hrsg. von S. Rinderle-Ma, F. Toumani und K. Wolf. Bd. 6896. Lecture Notes in Computer Science. Springer, S. 281–297. DOI: 10.1007/978-3-642-23059-2_22.

Belhoul, Y., M. Haddad, É. Duchêne und H. Kheddouci (2012). „String Comparators Based Algorithms for Process Model Matchmaking“. In: *9th International Conference on Services Computing (SCC), Honolulu, USA*. Hrsg. von L. E. Moser, M. Parashar und P. C. K. Hung. IEEE Computer Society, S. 649–656. DOI: 10.1109/SCC.2012.69.

Belhoul, Y., S. Yahiaoui, M. Haddad, A. Gater, H. Kheddouci und M. Bouzeghoub (2015). „A Graph Approach for Enhancing Process Models Matchmaking“. In: *2015 IEEE International Conference on Services Computing (SCC), New York City, USA*. IEEE Computer Society, S. 773–776. DOI: 10.1109/SCC.2015.112.

Bengio, Y., R. Ducharme, P. Vincent und C. Janvin (2003). „A Neural Probabilistic Language Model“. In: *Journal of Machine Learning Research* 3, S. 1137–1155.

Bergmann, R. und Y. Gil (2011). „Retrieval of Semantic Workflows with Knowledge Intensive Similarity Measures“. In: *19th International Conference on Case-Based Reasoning (ICCBR), London, England*. Hrsg. von A. Ram und N. Wiratunga. Bd. 6880. Lecture Notes in Computer Science. Springer, S. 17–31. DOI: 10.1007/978-3-642-23291-6_4.

Bergmann, R. und Y. Gil (2014). „Similarity Assessment and Efficient Retrieval of Semantic Workflows“. In: *Information Systems Journal* 40, S. 115–127. DOI: 10.1016/j.is.2012.07.005.

Bergmann, R., G. Müller und D. Wittkowsky (2013). „Workflow Clustering Using Semantic Similarity Measures“. In: *36th Annual German Conference on AI (KI), Koblenz, Deutschland*. Hrsg. von I. J. Timm und M. Thimm. Bd. 8077. Lecture Notes in Computer Science. Springer, S. 13–24. DOI: 10.1007/978-3-642-40942-4_2.

Berry, M. W., S. T. Dumais und G. W. O'Brien (1995). „Using Linear Algebra for Intelligent Information Retrieval“. In: *SIAM Review* 37.4, S. 573–595. DOI: 10.1137/1037127.

Blei, D. M., A. Y. Ng und M. I. Jordan (2003). „Latent Dirichlet Allocation“. In: *Journal of Machine Learning Research* 3, S. 993–1022.

Branco, M. C., J. Troya, K. Czarnecki, J. M. Küster und H. Völzer (2012). „Matching Business Process Workflows across Abstraction Levels“. In: *15th International Conference on Model Driven Engineering Languages and Systems (MODELS), Innsbruck, Österreich*. Hrsg. von R. B. France, J. Kazmeier, R. Breu und C. Atkinson. Bd. 7590. Lecture Notes in Computer Science. Springer, S. 626–641. DOI: 10.1007/978-3-642-33666-9_40.

Brockmans, S., M. Ehrig, A. Koschmider, A. Oberweis und R. Studer (2006). „Semantic Alignment Of Business Processes“. In: *8th International Conference on Enterprise Information Systems (ICEIS), Paphos, Zypern*. Hrsg. von Y. Manolopoulos, J. Filipe, P. Constantopoulos und J. Cordeiro. INSTICC PRESS, S. 191–196.

Business Process Model and Notation (BPMN) (01/2011). Object Management Group (OMG).

Cao, B., J. Wang, J. Fan, J. Yin und T. Dong (2016). „Querying Similar Process Models based on the Hungarian Algorithm“. In: *IEEE Transactions on Services Computing* 10.1, S. 121–135. DOI: 10.1109/TSC.2016.2597143.

Caporale, T. (2016). „A Tool for Natural Language Oriented Business Process Modeling“. In: *Central European Workshop on Services and their Composition (ZEUS), Wien, Österreich*. Hrsg. von C. Hochreiner und S. Schulte. Bd. 1562. CEUR Workshop Proceedings. CEUR, S. 49–52.

Carstensen, K.-U., C. Ebert, C. Ebert, S. Jekat, H. Langer und R. Klabunde, Hrsg. (2010). *Computerlinguistik und Sprachtechnologie - Eine Einführung*. 3. Aufl. Spektrum Akademischer Verlag, Wiesbaden. DOI: 10.1007/978-3-8274-2224-8.

Castano, S. und V. D. Antonellis (1995). „Reengineering Processes in Public Administration“. In: *14th International Conference on Object-Oriented and Entity-Relationship Modelling (OOER), Gold Coast, Australien*. Hrsg. von M. P. Papazoglou. Bd. 1021. Lecture Notes in Computer Science. Springer, S. 282–295. DOI: 10.1007/BFb0020540.

Castano, S. und V. D. Antonellis (1996). „Techniques for Process Analysis and Unification“. In: *8th International Conference on Advances Information System Engineering (CAiSE), Heraklion, Griechenland*. Hrsg. von P. Constantopoulos, J. Mylopoulos und Y. Vassiliou. Bd. 1080. Lecture Notes in Computer Science. Springer, S. 234–254. DOI: 10.1007/3-540-61292-0_14.

Cayoglu, U., R. Dijkman, M. Dumas, P. Fettke, L. García-Bañuelos, P. Hake, C. Klinkmüller, H. Leopold, A. Ludwig, P. Loos, J. Mendling, A. Oberweis, A. Schoknecht, E. Sheetrit, T. Thaler, M. Ullrich, I. Weber und M. Weidlich (2014). „Report: The Process Model Matching Contest 2013“. In: *Business Process Management Workshops, Peking, China*. Hrsg. von N. Lohmann, M. Song und P. Wohed. Bd. 171. Lecture Notes in Business Information Processing. Springer, S. 442–463. DOI: 10.1007/978-3-319-06257-0_35.

Chan, N. N., W. Gaaloul und S. Tata (2012). „Assisting Business Process Design by Activity Neighborhood Context Matching“. In: *10th International Conference on Service-Oriented Computing (ICSOC), Shanghai, China*. Hrsg. von C. Liu, H. Ludwig, F. Toumani und Q. Yu. Bd. 7636. Lecture Notes in Computer Science. Springer, S. 541–549. DOI: 10.1007/978-3-642-34321-6_38.

Choi, I., K. Kim und M. Jang (2007). „An XML-based process repository and process query language for integrated process management“. In: *Knowledge and Process Management* 14.4, S. 303–316. DOI: 10.1002/kpm.290.

Corrales, J. C., D. Grigori und M. Bouzeghoub (2006). „BPEL Processes Matchmaking for Service Discovery“. In: *On the Move to Meaningful Internet Systems (OTM), Confederated International Conferences CoopIS, DOA, GADA, and ODBASE, Part I, Montpellier, Frankreich*. Hrsg. von R. Meersman und Z. Tari. Bd. 4275. Lecture Notes in Computer Science. Springer, S. 237–254. DOI: 10.1007/11914853_15.

Cover, T. M. und J. A. Thomas (2006). *Elements of Information Theory*. 2. Aufl. Wiley, Hoboken, New Jersey, USA. DOI: 10.1002/047174882X.

Cuff, R. N. (1980). „On casual users“. In: *International Journal of Man-Machine Studies* 12.2, S. 163–187. DOI: 10.1016/S0020-7373(80)80016-2.

Curran, T. A. und A. Ladd (1999). *SAP R/3 Business Blueprint: Understanding Enterprise Supply Chain Management*. 2. Aufl. Prentice Hall PTR, Upper Saddle River, New Jersey, USA.

Curtis, B., M. I. Kellner und J. Over (1992). „Process Modeling“. In: *Communications of the ACM* 35.9, S. 75–90. DOI: 10.1145/130994.130998.

Davenport, T. H. (1993). *Process Innovation: Reengineering Work Through Information Technology*. Harvard Business School Press, Boston, Massachusetts, USA.

De Medeiros, A. K. A., W. M. P. van der Aalst und A. J. M. M. Weijters (2008). „Quantifying process equivalence based on observed behavior". In: *Data & Knowledge Engineering* 64.1, S. 55–74. DOI: 10.1016/j.datak.2007.06.010.

De Murillas, E. G. L., H. A. Reijers und W. M. P. van der Aalst (2017). „Everything You AlwaysWanted to Know About Your Process, But Did Not Know How To Ask". In: *Business Process Management Workshops: BPM 2016 International Workshops, Rio de Janeiro, Brasilien, Revised Papers*. Hrsg. von M. Dumas und M. Fantinato. Bd. 281. Lecture Notes in Business Information Processing. Springer, S. 296–309. DOI: 10.1007/978-3-319-58457-7_22.

Deerwester, S. C., S. T. Dumais, T. K. Landauer, G. W. Furnas und R. A. Harshman (1990). „Indexing by Latent Semantic Analysis". In: *Journal of the American Society for Information Science* 41.6, S. 391–407. DOI: 10.1002/(SICI)1097-4571(199009)41:6<391::AID-ASI1>3.0.CO;2-9.

Delfmann, P., M. Steinhorst, H.-A. Dietrich und J. Becker (2015). „The generic model query language GMQL — Conceptual specification, implementation, and runtime evaluation". In: *Information Systems* 47, S. 129–177. DOI: 10.1016/j.is.2014.06.003.

Desel, J. und G. Juhás (2001). „"What Is a Petri Net?" Informal Answers for the Informed Reader". In: *Unifying Petri Nets, Advances in Petri Nets*. Hrsg. von H. Ehrig, G. Juhás, J. Padberg und G. Rozenberg. Bd. 2128. Lecture Notes in Computer Science. Springer, S. 1–25. DOI: 10.1007/3-540-45541-8_1.

Dijkman, R. M., M. Dumas, L. García-Bañuelos und R. Käärik (2009a). „Aligning Business Process Models". In: *13th IEEE International Enterprise Distributed Object Computing Conference (EDOC), Auckland, Neuseeland*. IEEE Computer Society, S. 45–53. DOI: 10.1109/EDOC.2009.11.

Dijkman, R., M. Dumas und L. García-Bañuelos (2009b). „Graph Matching Algorithms for Business Process Model Similarity Search". In: *7th International Conference on Business Process Management (BPM), Ulm, Deutschland*. Hrsg. von U. Dayal, J. Eder, J. Koehler und H. A. Reijers. Bd. 5701. Lecture Notes in Computer Science. Springer, S. 48–63. DOI: 10.1007/978-3-642-03848-8_5.

Dijkman, R., B. Gfeller, J. Käster und H. Völzer (2011). „Identifying refactoring opportunities in process model repositories". In: *Information and Software Technology* 53.9, S. 937–948. DOI: 10.1016/j.infsof.2011.04.001.

Dijkstra, E. W. (1959). „A Note on Two Problems in Connexion with Graphs". In: *Numerische Mathematik* 1.1, S. 269–271. DOI: 10.1007%2FBF01386390.

Dumais, S. T. (1991). „Improving the retrieval of information from external sources". In: *Behavior Research Methods, Instruments, & Computers* 23.2, S. 229–236. DOI: 10.3758/BF03203370.

Dumais, S. T. (2007). „LSA and Information Retrieval: Getting Back to Basics". In: *Handbook of Latent Semantic Analysis*. Hrsg. von T. K. Landauer, D. S. McNamara, S. Dennis und W. Kintsch. Lawrence Erlbaum Associates, Inc., S. 293–321. DOI: 10.4324/9780203936399.ch16.

Dumas, M., L. García-Bañuelos und R. M. Dijkman (2009). „Similarity Search of Business Process Models". In: *IEEE Data Engineering Bulletin* 32.3, S. 23–28.

Dumas, M., L. García-Bañuelos, M. La Rosa und R. Uba (2013a). „Fast detection of exact clones in business process model repositories". In: *Information Systems Journal* 38.4, S. 619–633. DOI: 10.1016/j.is.2012.07.002.

Dumas, M., M. La Rosa, J. Mendling und H. A. Reijers (2013b). *Fundamentals of Business Process Management*. 1. Aufl. Springer, Berlin Heidelberg. DOI: 10.1007/978-3-642-33143-5.

Ehrig, M., A. Koschmider und A. Oberweis (2007). „Measuring Similarity between Semantic Business Process Models". In: *4th Asia-Pacific Conference on Conceptual Modelling (APCCM), Ballarat, Australia*. Hrsg. von J. F. Roddick und A. Hinze. Bd. 67. Conferences in Research and Practice in Information Technology. Australian Computer Society, S. 71–80.

Ekanayake, C. C., M. Dumas, L. García-Bañuelos, M. La Rosa und A. H. M. ter Hofstede (2012). „Approximate Clone Detection in Repositories of Business Process Models". In: *10th International Conference on Business Process Management (BPM), Tallinn, Estland*. Hrsg. von A. P. Barros, A. Gal und E. Kindler. Bd. 7481. Lecture Notes in Computer Science. Springer, S. 302–318. DOI: 10.1007/978-3-642-32885-5_24.

Elias, M. und P. Johannesson (2012). „A Survey of Process Model Reuse Repositories". In: *6th International Conference on Information Systems, Technology and Management (ICISTM), Grenoble, Frankreich*. Hrsg. von S. Dua, A. Gangopadhyay, P. Thulasiraman, U. Straccia, M. Shepherd und B. Stein. Bd. 285. Communications in Computer and Information Science. Springer, S. 64–76. DOI: 10.1007/978-3-642-29166-1_6.

Esgin, E. und P. Karagoz (2013a). „Confidence-Aware Sequence Alignment for Process Diagnostics". In: *International Conference on Signal-Image Technology & Internet-Based Systems (SITIS), Kyoto, Japan*. Hrsg. von K. Yétongnon, A. Dipanda und R. Chbeir. IEEE Computer Society, S. 990–997. DOI: 10.1109/SITIS.2013.160.

Esgin, E. und P. Karagoz (2013b). „Sequence Alignment Adaptation for Process Diagnostics and Delta Analysis". In: *8th International Conference on Hybrid Artificial Intelligent Systems (HAIS), Salamanca, Spanien*. Hrsg. von J. Pan, M. M. Polycarpou, M. Wozniak, A. C. P. L. F. Carvalho, H. Quintián-Pardo und E. Corchado. Bd. 8073. Lecture Notes in Computer Science. Springer, S. 191–201. DOI: 10.1007/978-3-642-40846-5_20.

Esgin, E. und P. Senkul (2011). „Delta Analysis: A Hybrid Quantitative Approach for Measuring Discrepancies between Business Process Models". In: *6th International Conference on Hybrid Artificial Intelligent Systems (HAIS), Part I, Wroclaw, Polen*. Hrsg. von E. Corchado, M. Kurzynski und M. Wozniak. Bd. 6678. Lecture Notes in Computer Science. Springer, S. 296–304. DOI: 10.1007/978-3-642-21219-2_38.

Eshuis, R. und P. W. P. J. Grefen (2007). „Structural Matching of BPEL Processes". In: *5th European Conference on Web Services (ECOWS), Halle (Saale), Deutschland*. IEEE Computer Society, S. 171–180. DOI: 10.1109/ECOWS.2007.26.

Euzenat, J. und P. Shvaiko (2013). *Ontology Matching*. 2. Aufl. Berlin Heidelberg: Springer. DOI: 10.1007/978-3-642-38721-0.

Evermann, J. (2009). „Theories of meaning in schema matching: An exploratory study". In: *Information Systems* 34.1, S. 28–44. DOI: 10.1016/j.is.2008.04.001.

Fahland, D., D. Lübke, J. Mendling, H. A. Reijers, B. Weber, M. Weidlich und S. Zugal (2009). „Declarative versus Imperative Process Modeling Languages: The Issue of Understandability". In: *10th International Workshop on Enterprise, Business-Process and Information Systems Modeling (BPMDS) and 14th International Conference on Exploring Modeling Methods for Systems Analysis and Design (EMMSAD), Amsterdam, Niederlande*. Hrsg. von T. A. Halpin, J. Krogstie, S. Nurcan, E. Proper, R. Schmidt, P. Soffer und R. Ukor. Bd. 29. Lecture Notes in Business Information Processing. Springer, S. 353–366. DOI: 10.1007/978-3-642-01862-6_29.

Fellbaum, C. (1998). *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, Massachusetts, USA.

Fellmann, M., P. Fettke, C. Houy, P. Loos, A. Oberweis, A. Schoknecht, M. Striewe, T. Thaler und M. Ullrich (2016). „Evaluation automatisierter Ansätze für die Bewertung von Modellierungsaufgaben“. In: *DeLFI 2016 – Die 14. E-Learning Fachtagung Informatik, Potsdam, Deutschland*. Hrsg. von U. Lucke, A. Schwill und R. Zender. Bd. P-262. Lecture Notes in Informatics. Gesellschaft für Informatik, S. 203–214.

Fellmann, M., A. Koschmider und A. Schoknecht (2014). „Analysis of Business Process Model Reuse Literature: Are Research Concepts Empirically Validated?“ In: *Modellierung 2014, Wien, Österreich*. Hrsg. von H. Fill, D. Karagiannis und U. Reimer. Bd. P-225. Lecture Notes in Informatics. Gesellschaft für Informatik, S. 185–192.

Fengel, J. (2014). „Semantic technologies for aligning heterogeneous business process models“. In: *Business Process Management Journal* 20.4, S. 549–570. DOI: 10.1108/BPMJ-07-2013-0085.

Fengel, J. und K. Reinking (2012). „Sprachbezogener Abgleich der Fachsemantik in heterogenen Geschäftsprozessmodellen“. In: *Modellierung 2012, Bamberg, Deutschland*. Hrsg. von E. J. Sinz und A. Schürr. Bd. P-201. Lecture Notes in Informatics. Gesellschaft für Informatik, S. 43–58.

Fill, H. und D. Karagiannis (2013). „On the Conceptualisation of Modelling Methods Using the ADOxx Meta Modelling Platform“. In: *Enterprise Modelling and Information Systems Architectures* 8.1, S. 4–25. DOI: 10.18417/emisa.8.1.1.

Francescomarino, C. D. und P. Tonella (2008). „Crosscutting Concern Documentation by Visual Query of Business Processes“. In: *Business Process Management Workshops, Mailand, Italien*. Hrsg. von D. Ardagna, M. Mecella und J. Yang. Bd. 17. Lecture Notes in Business Information Processing. Springer, S. 18–31. DOI: 10.1007/978-3-642-00328-8_3.

Fu, X., K. Yue, P. Zou, F. Wang und K. Ji (2012). „A Process Distance Metric Based on Alignment of Process Structure Trees“. In: *International Workshops on Web Technologies and Applications (APWeb Workshops), Kunming, China*. Hrsg. von H. Wang, L. Zou, G. Huang, J. He, C. Pang, H. L. Zhang, D. Zhao und Z. Yi. Bd. 7234. Lecture Notes in Computer Science. Springer, S. 221–232. DOI: 10.1007/978-3-642-29426-6.

Gacitua-Decar, V. und C. Pahl (2009). „Automatic Business Process Pattern Matching for Enterprise Services Design“. In: *2009 IEEE World Congress on Services (SERVICES), Part II, Bangalore, Indien*. IEEE Computer Society, S. 111–118. DOI: 10.1109/SERVICES-2.2009.28.

Gacitua-Decar, V. und C. Pahl (2010). „Towards Reuse of Business Processes Patterns to Design Services“. In: *Emerging Web Services Technology Volume III, Dublin, Irland*. Hrsg. von W. Binder und S. Dustdar. Whitestein Series in Software Agent Technologies and Autonomic Computing. Birkhäuser Basel, S. 15–36. DOI: 10.1007/978-3-0346-0104-7_2.

Gao, J. und L. Zhang (2009). „On Measuring Semantic Similarity of Business Process Models“. In: *International Conference on Interoperability for Enterprise Software and Applications (IESA), Peking, China*. IEEE Computer Society, S. 289–293. DOI: 10.1109/I-ESA.2009.50.

Gao, J., L. Zhang und W. Jiang (2007). „Procuring Requirements for ERP Software Based on Semantic Similarity“. In: *1st International Conference on Semantic Computing (ICSC), Irvine, USA*. IEEE Computer Society, S. 61–70. DOI: 10.1109/ICSC.2007.45.

Gao, X., Y. Chen, Z. Ding, M. Wang, X. Zhang, Z. Yan, L. Wen, Q. Guo und R. Chen (2013). „Process Model Fragmentization, Clustering and Merging: An Empirical Study“. In: *Business Process Management Workshops, Peking, China*. Hrsg. von N. Lohmann, M. Song und P. Wohed. Bd. 171. Lecture Notes in Business Information Processing. Springer, S. 405–416. DOI: 10.1007/978-3-319-06257-0_32.

Gater, A., D. Grigori und M. Bouzeghoub (2010). „Complex Mapping Discovery for Semantic Process Model Alignment“. In: *12th International Conference on Information Integration and Web-based Applications and Services (iiWAS), Paris, Frankreich*. Hrsg. von G. Kotsis, D. Taniar, E. Pardede, I. Saleh und I. Khalil. ACM, S. 317–324. DOI: 10.1145/1967486.1967537.

Gater, A., D. Grigori und M. Bouzeghoub (2012). „Indexing Process Model Flow Dependencies for Similarity Search“. In: *On the Move to Meaningful Internet Systems (OTM), Confederated International Conferences CoopIS, DOA-SVI, and ODBASE, Part I, Rom, Italien*. Hrsg. von R. Meersman, H. Panetto, T. S. Dillon, S. Rinderle-Ma, P. Dadam, X. Zhou, S. Pearson, A. Ferscha, S. Bergamaschi und I. F. Cruz. Bd. 7565. Lecture Notes in Computer Science. Springe, S. 128–145. DOI: 10.1007/978-3-642-33606-5_9.

Gater, A., D. Grigori, M. Haddad, M. Bouzeghoub und H. Kheddouci (2011). „A Summary-Based Approach for Enhancing Process Model Matchmaking“. In: *2011 IEEE International Conference on Service-Oriented Computing and Applications (SOCA), Irvine,USA*. Hrsg. von K. Lin, C. Huemer, M. B. Blake und B. Benatallah. IEEE Computer Society, S. 1–8. DOI: 10.1109/SOCA.2011.6166210.

Gerke, K., J. Cardoso und A. Claus (2009). „Measuring the Compliance of Processes with Reference Models“. In: *On the Move to Meaningful Internet Systems (OTM), Confederated International Conferences, CoopIS, DOA, IS, and ODBASE, Part I, Vilamoura, Portugal*. Hrsg. von R. Meersman, T. Dillon und P. Herrero. Bd. 5870. Lecture Notes in Computer Science. Springer, S. 76–93. DOI: 10.1007/978-3-642-05148-7_8.

Gerth, C., M. Luckey, J. Küster und G. Engels (2010). „Detection of Semantically Equivalent Fragments for Business Process Model Change Management“. In: *7th International Conference on Services Computing (SCC), Miami, USA*. IEEE Computer Society, S. 57–64. DOI: 10.1109/SCC.2010.38.

Gerth, C., M. Luckey, J. Küster und G. Engels (2011). „Precise Mappings between Business Process Models in Versioning Scenarios“. In: *8th International Conference on Services Computing (SCC), Washington, DC, USA*. Hrsg. von H. Jacobsen, Y. Wang und P. Hung. IEEE Computer Society, S. 218–225. DOI: 10.1109/SCC.2011.65.

Goud, R., K. M. van Hee, R. D. J. Post und J. M. E. M. van der Werf (2006). „Petriweb: A Repository for Petri Nets“. In: *27th International Conference on Applications and Theory of Petri Nets and Other Models of Concurrency (ICATPN), Turku, Finnland*. Hrsg. von S. Donatelli und P. S. Thiagarajan. Bd. 4024. Lecture Notes in Computer Science. Springer, S. 411–420. DOI: 10.1007/11767589_24.

Greene, S. L., L. M. Gomez und S. J. Devlin (1986). „A Cognitive Analysis of Database Query Production“. In: *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*. Bd. 30. 1. SAGE, S. 9–13. DOI: 10.1177/154193128603000103.

Grigori, D., J. C. Corrales und M. Bouzeghoub (2006). „Behavioral matchmaking for service retrieval“. In: *International Conference on Web Services (ICWS), Chicago, USA*. IEEE Computer Society, S. 145–152. DOI: 10.1109/ICWS.2006.37.

Grigori, D., J. C. Corrales und M. Bouzeghoub (2008). „Behavioral matchmaking for service retrieval: Application to conversation protocols“. In: *Information Systems – Advances in Data and Service Integration* 33.7–8, S. 681–698. DOI: 10.1016/j.is.2008.02.004.

Grigori, D., J. C. Corrales, M. Bouzeghoub und A. Gater (2010). „Ranking BPEL Processes for Service Discovery“. In: *IEEE Transactions on Services Computing* 3.3, S. 178–192. DOI: 10.1109/TSC.2010.6.

Haisjackl, C., S. Zugal, P. Soffer, I. Hadar, M. Reichert, J. Pinggera und B. Weber (2013). „Making Sense of Declarative Process Models: Common Strategies and Typical Pitfalls“. In: *14th International Conference on Enterprise, Business-Process and Information Systems Modeling (BPMDS), 18th International EMMSAD Conference, Valencia, Spanien*. Hrsg. von S. Nurcan, H. A. Proper, P. Soffer, J. Krogstie, R. Schmidt, T. A. Halpin und I. Bider. Bd. 147. Lecture Notes in Business Information Processing. Springer, S. 2–17. DOI: 10.1007/978-3-642-38484-4_2.

Hammer, M. und J. Champy (1994). *Reengineering the Corporation: A Manifesto for Business Revolution*. HarperBusiness, New York, USA.

Harmon, P. und C. Wolf (2011). *Business Process Modeling Survey*. Techn. Ber. Business Process Trends.

Harmon, P. und C. Wolf (2014). *The State of Business Process Management 2014*. Techn. Ber. Business Process Trends.

Hidders, J., M. Dumas, W. M. P. van der Aalst, A. H. M. ter Hofstede und J. Verelst (2005). „When are two Workflow Processes the same?“ In: *11th Australasian Theory Symposium (CATS), Newcastle, Australien*. Hrsg. von M. D. Atkinson und F. K. H. A. Dehne. Bd. 41. Conferences in Research and Practice in Information Technology. Australian Computer Society, S. 3–11.

Hollenbach, C. und W. Frakes (1996). „Software process reuse in an industrial setting“. In: *4th International Conference on Software Reuse, Orlando, USA*. Hrsg. von M. Sitaraman. IEEE Computer Society, S. 22–30. DOI: 10.1109/ICSR.1996.496110.

Holmes, M. P., A. G. Gray und C. L. I. Jr. (2008). „QUIC-SVD: Fast SVD Using Cosine Trees“. In: *22nd Annual Conference on Neural Information Processing Systems: Advances in Neural Information Processing Systems, Vancouver, Kanada*. Hrsg. von D. Koller, D. Schuurmans, Y. Bengio und L. Bottou. Curran Associates, Inc., S. 673–680.

Hoppenbrouwers, S. J. B. A., H. A. (Proper und T. P. van der Weide (2005). „A Fundamental View on the Process of Conceptual Modeling“. In: *24th International Conference on Conceptual Modeling (ER), Klagenfurt, Österreich*. Hrsg. von L. Delcambre, C. Kop, H. Mayr, J. Mylopoulos und O. Pastor. Bd. 3716. Lecture Notes in Computer Science. Springer, S. 128–143. DOI: 10.1007/11568322_9.

Houy, C., P. Fettke, P. Loos, W. M. P. van der Aalst und J. Krogstie (2010). „BPM-in-the-Large – Towards a Higher Level of Abstraction in Business Process Management“. In: *E-Government, E-Services and Global Processes, Brisbane, Australien*. Hrsg. von M. Janssen, W. Lamersdorf, J. Pries-Heje und M. Rosemann. Bd. 334. IFIP Advances in Information and Communication Technology. Springer, S. 233–244. DOI: 10.1007/978-3-642-15346-4_19.

Hu, X., Z. Cai, P. Wiemer-Hastings, A. C. Graesser und D. S. McNamara (2007). „Strenghts, Limitations, and Extensions of LSA“. In: *Handbook of Latent Semantic Analysis*. Hrsg. von T. K. Landauer, D. S. McNamara, S. Dennis und W. Kintsch. Lawrence Erlbaum Associates, Inc., S. 401–425. DOI: 10.4324/9780203936399.ch20.

Huang, K., Z. Zhou, Y. Han, G. Li und J. Wang (2004). „An Algorithm for Calculating Process Similarity to Cluster Open-Source Process Designs“. In: *Grid and Cooperative Computing Workshops (GCC), Wuhan, China*. Hrsg. von H. Jin, Y. Pan, N. Xiao und J. Sun. Bd. 3252. Lecture Notes in Computer Science. Springer, S. 107–114. DOI: 10.1007/978-3-540-30207-0_14.

Huang, Z., J. Huai, H. Sun, X. Liu und X. Li (2009). „BestRec: A Behavior Similarity Based Approach to Services Recommendation“. In: *Congress on Services (SERVICES), Part I, Los Angeles, USA*. IEEE Computer Society, S. 46–53. DOI: 10.1109/SERVICES-I.2009.45.

Humm, B. G. und J. Fengel (2012). „Semantics-Based Business Process Model Similarity“. In: *15th International Conference on Business Information Systems (BIS), Vilnius, Litauen*. Hrsg. von W. Abramowicz, D. Kriksciuniene und V. Sakalauskas. Bd. 117. Lecture Notes in Business Information Processing. Springer, S. 36–47. DOI: 10.1007/978-3-642-30359-3_4.

Indulska, M., P. Green, J. Recker und M. Rosemann (2009). „Business Process Modeling: Perceived Benefits“. In: *28th International Conference on Conceptual Modeling (ER), Gramado, Brasilien*. Hrsg. von A. H. Laender, S. Castano, U. Dayal, F. Casati und J. P. M. de Oliveira. Bd. 5829. Lecture Notes in Computer Science. Springer, S. 458–471. DOI: 10.1007/978-3-642-04840-1_34.

Jin, T., J. Wang und L. Wen (2011). „Querying Business Process Models Based on Semantics“. In: *6th International Conference on Database Systems for Advanced Applications (DASFAA), Part II, Hong Kong, China*. Hrsg. von J. X. Yu, M.-H. Kim und R. Unland. Bd. 6588. Lecture Notes in Computer Science. Springer, S. 164–178. DOI: 10.1007/978-3-642-20152-3_13.

Jin, T., J. Wang und L. Wen (2012). „Efficient Retrieval of Similar Workflow Models Based on Behavior“. In: *14th Asia-Pacific Web Conference on Web Technologies and Applications (APWeb), Kunming, China*. Hrsg. von Q. Z. Sheng, G. Wang, C. S. Jensen und G. Xu. Bd. 7235. Lecture Notes in Computer Science. Springer, S. 677–684. DOI: 10.1007/978-3-642-29253-8.

Jivani, A. G. (2011). „A Comparative Study of Stemming Algorithms“. In: *International Journal of Computer Technology and Applications* 2, S. 1930–1938.

Jung, J.-Y. und J. Bae (2006). „Workflow Clustering Method Based on Process Similarity“. In: *International Conference on Computational Science and Its Applications (ICCSA), Part II, Glasgow, Schottland*. Hrsg. von M. L. Gavrilova, O. Gervasi, V. Kumar, C. J. K. Tan, D. Taniar, A. Lagana, Y. Mun und H. Choo. Bd. 3981. Lecture Notes in Computer Science. Springer, S. 379–389. DOI: 10.1007/11751588_40.

Jung, J.-Y., J. Bae und L. Liu (2009). „Hierarchical Clustering of Business Process Models“. In: *International Journal of Innovative Computing, Information and Control* 5.12(A), S. 4501–4511.

Jurafsky, D. und J. H. Martin (2009). *Speech and Language Processing: An Introduction to Natural Language Processing, Speech Recognition, and Computational Linguistics*. 2. Aufl. Prentice Hall series in artificial intelligence. Prentice Hall, Upper Saddle River, New Jersey, USA.

Karagiannis, D. und H. Kühn (2002). „Metamodelling Platforms“. In: *3d International Conference on E-Commerce and Web Technologies (EC-Web), Aix-en-Provence, Frankreich*. Hrsg. von K. Bauknecht, A. M. Tjoa und G. Quirchmayr. Bd. 2455. Lecture Notes in Computer Science. Springer, S. 182. DOI: 10.1007/3-540-45705-4_19.

Kastner, M., M. W. Saleh, S. Wagner, M. Affenzeller und W. Jacak (2009). „Heuristic Methods for Searching and Clustering Hierarchical Workflows“. In: *Revised Selected Papers of the 12th International Conference on Computer Aided Systems Theory (EUROCAST), Las Palmas de Gran Canaria, Spanien*. Hrsg. von R. Moreno-Díaz, F. Pichler und A. Quesada-Arencibia. Bd. 5717. Lecture Notes in Computer Science. Springer, S. 737–744. DOI: 10.1007/978-3-642-04772-5.

Keller, G., M. Nüttgens und A.-W. Scheer (1992). *Semantische Prozessmodellierung auf der Grundlage Ereignisgesteuerter Prozessketten (EPK)*. Techn. Ber. 89. Institut für Wirtschaftsinformatik.

Kesari, M., S. Chang und P. B. Seddon (2003). „A content-analytic study of the advantages and disadvantages of process modelling“. In: *14th Australasian Conference on Information Systems (ACIS), Perth, Australien*.

Kintsch, W., D. S. McNamara, S. Dennis und T. K. Landauer (2007). „LSA and Meaning: In Theory and Application“. In: *Handbook of Latent Semantic Analysis*. Hrsg. von T. K. Landauer, D. S. McNamara, S. Dennis und W. Kintsch. Lawrence Erlbaum Associates, S. 467–479. DOI: 10.4324/9780203936399.ch23.

Klein, D. und C. D. Manning (2003). „Accurate Unlexicalized Parsing“. In: *41st Annual Meeting of the Association for Computational Linguistics, Sapporo, Japan*. Hrsg. von E. W. Hinrichs und D. Roth. ACL, S. 423–430. DOI: 10.3115/1075096.1075150.

Klein, M. und C. Petti (2006). „A handbook-based methodology for redesigning business processes“. In: *Knowledge and Process Management* 13.2, S. 108–119. DOI: 10.1002/kpm.248.

Klinkmüller, C., H. Leopold, I. Weber, J. Mendling und A. Ludwig (2014). „Listen to Me: Improving Process Model Matching through User Feedback“. In: *12th International Conference on Business Process Management (BPM), Haifa, Israel*. Hrsg. von S. Sadiq, P. Soffer und H. Völzer. Bd. 8659. Lecture Notes in Computer Science. Springer, S. 84–100. DOI: 10.1007/978-3-319-10172-9_6.

Klinkmüller, C. und I. Weber (2017). „Analyzing control flow information to improve the effectiveness of process model matching techniques“. In: *Decision Support Systems* 100, S. 6–14. DOI: 10.1016/j.dss.2017.06.002.

Klinkmüller, C., I. Weber, J. Mendling, H. Leopold und A. Ludwig (2013). „Increasing Recall of Process Model Matching by Improved Activity Label Matching“. In: *11th International Conference on Business Process Management (BPM), Peking, China*. Hrsg. von F. Daniel, J. Wang und B. Weber. Bd. 8094. Lecture Notes in Computer Science. Springer, S. 211–218. DOI: 10.1007/978-3-642-40176-3_17.

Koch, S. (2011). *Einführung in das Management von Geschäftsprozessen*. Springer, Berlin Heidelberg. DOI: 10.1007/978-3-642-01121-4.

Kohavi, R. (1995). „A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection“. In: *14th International Joint Conference on Artificial Intelligence (IJCAI), Montréal, Kanada*. Morgan Kaufmann Publishers, S. 1137–1145.

Kohlbacher, M. (2010). „The effects of process orientation: a literature review“. In: *Business Process Management Journal* 16.1, S. 135–152. DOI: 10.1108/14637151011017985.

Koschmider, A., M. Fellmann, A. Schoknecht und A. Oberweis (2014). „Analysis of process model reuse: Where are we now, where should we go from here?“ In: *Decision Support Systems* 66, S. 9–19. DOI: 10.1016/j.dss.2014.05.012.

Koschmider, A., K. Figl und A. Schoknecht (2015). „A Comprehensive Overview of Visual Design of Process Model Element Labels“. In: *Business Process Management Workshops, Innsbruck, Österreich*. Hrsg. von M. Reichert und H. Reijers. Bd. 256. Lecture Notes in Business Information Processing. Springer, S. 571–582. DOI: 10.1007/978-3-319-42887-1_46.

Koschmider, A., T. Hornung und A. Oberweis (2011). „Recommendation-based editor for business process modeling“. In: *Data & Knowledge Engineering* 70.6, S. 483–503. DOI: 10.1016/j.datak.2011.02.002.

Koschmider, A. und A. Oberweis (2007). „How to detect semantic business process model variants?“ In: *ACM Symposium on Applied Computing (SAC), Seoul, Korea*. Hrsg. von Y. Cho, R. L. Wainwright, H. Haddad, S. Y. Shin und Y. W. Koo. Australian Computer Society, S. 1263–1264. DOI: 10.1145/1244002.1244274.

Kunze, M., M. Weidlich und M. Weske (2011a). „Behavioral Similarity – A Proper Metric“. In: *9th International Conference on Business Process Management (BPM), Clermont-Ferrand, Frankreich*. Hrsg. von S. Rinderle-Ma, F. Toumani und K. Wolf. Bd. 6896. Lecture Notes in Computer Science. Springer, S. 166–181. DOI: 10.1007/978-3-642-23059-2.

Kunze, M., M. Weidlich und M. Weske (2011b). „m3 – A Behavioral Similarity Metric for Business Processes“. In: *3rd Central-European Workshop on Services and their Composition (ZEUS), Karlsruhe, Deutschland*. Hrsg. von D. Eichhorn, A. Koschmider und H. Zhang. Bd. 705. CEUR Workshop Proceedings. CEUR, S. 89–95.

Kunze, M., M. Weidlich und M. Weske (2013). „Querying process models by behavior inclusion“. In: *Software & Systems Modeling* 14.3, S. 1–21. DOI: 10.1007/s10270-013-0389-6.

Kunze, M. und M. Weske (2010). „Metric Trees for Efficient Similarity Search in Large Process Model Repositories“. In: *Business Process Management Workshops, Hoboken, USA*. Hrsg. von M. zur Muehlen und J. Su. Bd. 66. Lecture Notes in Business Information Processing. Springer, S. 535–546. DOI: 10.1007/978-3-642-20511-8.

Kunze, M. und M. Weske (2012). „Local Behavior Similarity“. In: *13th International Conference on Enterprise, Business-Process and Information Systems Modeling and 17th International Conference on Exploring Modelling Methods for Systems Analysis and Design (BMMDS/EMMSAD), Gdańsk, Polen*. Hrsg. von I. Bider, T. A. Halpin, J. Krogstie, S. Nurcan, E. Proper, R. Schmidt, P. Soffer und S. Wrycza. Bd. 113. Lecture Notes in Business Information Processing. Springer, S. 107–120. DOI: 10.1007/978-3-642-31072-0.

Kusner, M. J., Y. Sun, N. I. Kolkin und K. Q. Weinberger (2015). „From Word Embeddings To Document Distances“. In: *32nd International Conference on Machine Learning (ICML), Lille, Frankreich*. Hrsg. von F. R. Bach und D. M. Blei. Bd. 37. JMLR Workshop and Conference Proceedings. JMLR, S. 957–966.

Kuss, E., H. Leopold, H. van der Aa, H. Stuckenschmidt und H. A. Reijers (2016). „Probabilistic Evaluation of Process Model Matching Techniques“. In: *35th International Conference on Conceptual Modeling (ER), Gifu, Japan*. Hrsg. von I. Comyn-Wattiau, K. Tanaka, I. Song, S. Yamamoto und M. Saeki. Bd. 9974. Lecture Notes in Computer Science. Springer, S. 279–292. DOI: 10.1007/978-3-319-46397-1_22.

Küster, J. M., J. Koehler und K. Ryndina (2006). „Improving Business Process Models with Reference Models in Business-Driven Development“. In: *Business Process Management Workshops, Wien, Österreich*. Hrsg. von J. Eder und S. Dustdar. Bd. 4103. Lecture Notes in Computer Science. Springer, S. 35–44. DOI: 10.1007/11837862_5.

Küster, J. M., C. Gerth, A. Förster und G. Engels (2008). „Detecting and Resolving Process Model Differences in the Absence of a Change Log“. In: *6th International Conference on Business Process Management (BPM), Mailand, Italien*. Hrsg. von M. Dumas, M. Reichert und M. Shan. Bd. 5240. Lecture Notes in Computer Science. Springer, S. 244–260. DOI: 10.1007/978-3-540-85758-7_19.

La Rosa, M., M. Dumas, C. C. Ekanayake, L. García-Bañuelos, J. Recker und A. H. M. ter Hofstede (2015). „Detecting approximate clones in business process model repositories“. In: *Information Systems* 49, S. 102–125. DOI: 10.1016/j.is.2014.11.010.

La Rosa, M., M. Dumas, R. Uba und R. M. Dijkman (2010). „Merging Business Process Models“. In: *On the Move to Meaningful Internet Systems (OTM), Confederated International Conferences CoopIS, IS, DOA and ODBASE, Part I, Hersonissos, Griechenland*. Hrsg. von R. Meersman, T. Dillon und P. Herrero. Bd. 6426. Lecture Notes in Computer Science. Springer, S. 96–113. DOI: 10.1007/978-3-642-16934-2_10.

La Rosa, M., M. Dumas, R. Uba und R. M. Dijkman (2013). „Business Process Model Merging: An Approach to Business Process Consolidation“. In: *ACM Transactions on Software Engineering and Methodology (TOSEM)* 22.2, 11:1–11:42. DOI: 10.1145/2430545.2430547.

Lam, V. S. W. (2009). „Equivalences of BPMN processes“. In: *Service Oriented Computing and Applications* 3.3, S. 189–204. DOI: 10.1007/s11761-009-0048-5.

Landauer, T. K., P. W. Foltz und D. Laham (1998). „An introduction to Latent Semantic Analysis“. In: *Discourse Processes* 25.2-3, S. 259–284. DOI: 10.1080/01638539809545028.

Landauer, T. K. (2007). „LSA as a Theory of Meaning“. In: *Handbook of Latent Semantic Analysis*. Hrsg. von T. K. Landauer, D. S. McNamara, S. Dennis und W. Kintsch. Lawrence Erlbaum Associates, S. 3–34. DOI: 10.4324/9780203936399.ch1.

Lau, C. K., A. J. Fournier, Y. Xia, J. Recker und E. Bernhard (2011). *Process Model Repository Governance at Suncorp*. Techn. Ber. Business Process Management Research Group, Queensland University of Technology.

Laue, R. und M. Becker (2012). „Evaluating Social Tagging for Business Process Models“. In: *Business Process Management Workshops, Tallinn, Estland*. Hrsg. von M. La Rosa und P. Soffer. Bd. 132. Lecture Notes in Business Information Processing. Springer, S. 280–291. DOI: 10.1007/978-3-642-36285-9_33.

Leopold, H. (2013). *Natural Language in Business Process Models - Theoretical Foundations, Techniques, and Applications*. Bd. 168. Lecture Notes in Business Information Processing. Springer, Cham, Schweiz. DOI: 10.1007/978-3-319-04175-9.

Leopold, H., M. Niepert, M. Weidlich, J. Mendling, R. M. Dijkman und H. Stuckenschmidt (2012). „Probabilistic Optimization of Semantic Process Model Matching“. In: *10th International Conference on Business Process Management (BPM), Tallinn, Estland*. Hrsg. von A. P. Barros, A. Gal und E. Kindler. Bd. 7481. Lecture Notes in Computer Science. Springer, S. 319–334. DOI: 10.1007/978-3-642-32885-5_25.

Leopold, H., H. van der Aa, F. Pittke, M. Raffel, J. Mendling und H. A. Reijers (2016). „Integrating Textual and Model-Based Process Descriptions for Comprehensive Process Search“. In: *17th International Conference on Enterprise, Business-Process and Information Systems Modeling (BPMDS), Ljubljana, Slovenien*. Hrsg. von R. Schmidt, W. Guédria, I. Bider und S. Guerreiro. Bd. 248. Lecture Notes in Business Information Processing. Springer, S. 51–65. DOI: 10.1007/978-3-319-39429-9_4.

Levenshtein, V. I. (1966). „Binary codes capable of correcting deletions, insertions, and reversals“. In: *Soviet Physics Doklady* 10.9, S. 707–710.

Lewis, D. D., Y. Yang, T. Rose und F. Li (2004). „RCV1: A New Benchmark Collection for Text Categorization Research“. In: *Journal of Machine Learning Research* 5, S. 361–397.

Li, C., M. Reichert und A. Wombacher (2008). „On Measuring Process Model Similarity Based on High-Level Change Operations“. In: *27th International Conference on Conceptual Modeling (ER), Barcelona, Spanien*. Hrsg. von Q. Li, S. Spaccapietra, E. Yu und A. Olivé. Bd. 5231. Lecture Notes in Computer Science. Springer, S. 248–264. DOI: 10.1007/978-3-540-87877-3_19.

Li, C., M. Reichert und A. Wombacher (2010). „The Minadept Clustering Approach for Discovering Reference Process Models Out of Process Variants“. In: *International Journal of Cooperative Information Systems* 19.3-4, S. 159–203. DOI: 10.1142/S0218843010002139.

Lin, D. (1998). „An Information-Theoretic Definition of Similarity“. In: *15th International Conference on Machine Learning (ICML), Madison, USA*. Hrsg. von J. W. Shavlik. Morgan Kaufmann, S. 296–304.

Lincoln, M. und A. Gal (2011). „Searching Business Process Repositories Using Operational Similarity“. In: *On the Move to Meaningful Internet Systems (OTM), Confederated International Conferences CoopIS, DOA-SVI, and ODBASE, Hersonissos, Griechenland*. Hrsg. von R. Meersman, T. S. Dillon, P. Herrero, A. Kumar, M. Reichert, L. Qing, B. C. Ooi, E. Damiani, D. C. Schmidt, J. White, M. Hauswirth, P. Hitzler und M. K. Mohania. Bd. 7044. Lecture Notes in Computer Science. Springer, S. 2–19. DOI: 10.1007/978-3-642-25109-2_2.

Ling, J., L. Zhang und Q. Feng (2014). „Business Process Model Alignment: An Approach to Support Fast Discovering Complex Matches“. In: *I-ESA Conferences Enterprise Interoperability VI: Interoperability for Agility, Resilience and Plasticity of Collaborations, Albi, Frankreich*. Hrsg. von K. Mertins, F. Benaben, R. Poler und J.-P. Bourreres. Bd. 7. I-ESA Conferences. Springer, S. 41–51. DOI: 10.1007/978-3-319-04948-9_4.

Lipton, Z. C., C. Elkan und B. Naryanaswamy (2014). „Optimal Thresholding of Classifiers to Maximize F1 Measure“. In: *Machine Learning and Knowledge Discovery in Databases, European Conference, ECML PKDD, Part II, Nancy, Frankreich*. Hrsg. von T. Calders, F. Esposito, E. Hüllermeier und R. Meo. Bd. 8725. Lecture Notes in Computer Science. Springer, S. 225–239. DOI: 10.1007/978-3-662-44851-9_15.

Liu, H., G. Liu, Y. Wang und D. Liu (2012). „A Novel Behavioral Similarity Measure for Artifact-Oriented Business Processes“. In: *Technology for Education and Learning, Macau, China*. Hrsg. von H. Tan. Bd. 136. Advances in Intelligent Systems and Computing. Springer, S. 81–88. DOI: 10.1007/978-3-642-27711-5_12.

Liu, K., Z. Yan, Y. Wang, L. Wen und J. Wang (2014). „Efficient Syntactic Process Difference Detection Using Flexible Feature Matching“. In: *2nd Asia Pacific Business Process Management Conference (AP-BPM), Brisbane, Australien*. Hrsg. von C. Ouyang und J.-Y. Jung. Bd. 181. Lecture Notes in Business Information Processing. Springer, S. 103–116. DOI: 10.1007/978-3-319-08222-6_8.

Liu, Z., Q. Shao und Y. Chen (2010). „Searching Workflows with Hierarchical Views“. In: *Proceedings of the VLDB Endowment* 3.1, S. 918–927. DOI: 10.14778/1920841.1920958.

Lu, R. und S. W. Sadiq (2007). „On the Discovery of Preferred Work Practice Through Business Process Variants“. In: *26th International Conference on Conceptual Modeling (ER), Auckland, Neuseeland*. Hrsg. von C. Parent, K.-D. Schewe, V. C. Storey und B. Thalheim. Bd. 4801. Lecture Notes in Computer Science. Springer, S. 165–180. DOI: 10.1007/978-3-540-75563-0_13.

Lu, R., S. Sadiq und G. Governatori (2009). „On managing business processes variants“. In: *Data & Knowledge Engineering* 68.7, S. 642–664. DOI: 10.1016/j.datak.2009.02.009.

Lübbe, A. (2011). „Tangible Business Process Modeling: Design and Evaluation of a Process Model Elicitation Technique“. Diss. University of Potsdam.

Luhn, H. P. (1960). „Key word-in-context index for technical literature (kwic index)“. In: *American Documentation* 11.4, S. 288–295. DOI: 10.1002/asi.5090110403.

Lv, Y. und C. Zhai (2009). „Positional language models for information retrieval“. In: *32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR), Boston, USA*. Hrsg. von J. Allan, J. A. Aslam, M. Sanderson, C. Zhai und J. Zobel. ACM, S. 299–306. DOI: 10.1145/1571941.1571994.

Madhusudan, T., J. L. Zhao und B. Marshall (2004). „A case-based reasoning framework for workflow model management“. In: *Data & Knowledge Engineering* 50.1, S. 87–115. DOI: 10.1016/j.datak.2004.01.005.

Mahmod, N. M. und W. Y. Chiew (2010). „Structural Similarity of Business Process Variants“. In: *Conference on Open Systems (ICOS), Kuala Lumpur, Malaysia*. IEEE Computer Society, S. 17–22. DOI: 10.1109/ICOS.2010.5720057.

Mahmod, N. M. und S. b. A. Radzi (2010). „An Approach to Analyse Similarity of Business Process Variants“. In: *International Conference on Progress in Informatics and Computing (PIC), Shanghai, China*. IEEE Computer Society, S. 640–644. DOI: 10.1109/PIC.2010.5687872.

Malinova, M., R. M. Dijkman und J. Mendling (2013). „Automatic Extraction of Process Categories from Process Model Collections“. In: *Business Process Management Workshops, Peking, China*. Hrsg. von N. Lohmann, M. Song und P. Wohed. Bd. 171. Lecture Notes in Business Information Processing. Springer, S. 430–441. DOI: 10.1007/978-3-319-06257-0_34.

Manning, C. D., P. Raghavan und H. Schütze (2008). *Introduction to Information Retrieval*. Cambridge University Press, Cambridge, England. DOI: 10.1017/CBO9780511809071.

Markovic, I. (2008). „Advanced Querying and Reasoning on Business Process Models“. In: *11th International Conference on Business Information Systems (BIS), Innsbruck, Österreich*. Hrsg. von W. Abramowicz und D. Fensel. Bd. 7. Lecture Notes in Business Information Processing. Springer, S. 189–200. DOI: 10.1007/978-3-540-79396-0_17.

Markovic, I., A. C. Pereira, D. de Francisco Marcos und H. Muñoz (2007). „Querying in Business Process Modeling“. In: *Service-Oriented Computing Workshops (ICSOC), Revised Selected Papers, Wien, Österreich*. Hrsg. von E. D. Nitto und M. Ripeanu. Bd. 4907. Lecture Notes in Computer Science. Springer, S. 234–245. DOI: 10.1007/978-3-540-93851-4_23.

Martens, A., P. Fettke und P. Loos (2014). „A Genetic Algorithm for the Inductive Derivation of Reference Models Using Minimal Graph-Edit Distance Applied to Real-World Business Process Data“. In: *Multikonferenz Wirtschaftsinformatik (MKWI), Paderborn, Deutschland*. Hrsg. von D. Kundisch, L. Suhl und L. Beckmann. Universität Paderborn, S. 1613–1626.

Martin, D. I. und M. W. Berry (2007). „Mathematical foundations behind latent semantic analysis“. In: *Handbook of Latent Semantic Analysis*. Hrsg. von T. K. Landauer, D. S. McNamara, S. Dennis und W. Kintsch. Lawrence Erlbaum Associates, S. 35–56. DOI: 10.4324/9780203936399.ch2.

Meilicke, C., H. Leopold, E. Kuss, H. Stuckenschmidt und H. A. Reijers (2017). „Overcoming individual process model matcher weaknesses using ensemble matching“. In: *Decision Support Systems* 100, S. 15–26. DOI: 10.1016/j.dss.2017.02.013.

Melcher, J. und D. Seese (2008). „Visualization and Clustering of Business Process Collections Based on Process Metric Values“. In: *10th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC), Timisoara, Rumänien*. Hrsg. von V. Negru, T. Jebelean, D. Petcu und D. Zaharie. IEEE Computer Society, S. 572–575. DOI: 10.1109/SYNASC.2008.37.

Mendling, J., G. Neumann und W. M. P. van der Aalst (2007a). „Understanding the Occurrence of Errors in Process Models Based on Metrics“. In: *On the Move to Meaningful Internet Systems (OTM), Confederated International Conferences CoopIS, DOA, ODBASE, GADA, and IS, Vilamoura, Portugal*. Hrsg. von R. Meersman und Z. Tari. Bd. 4803. Lecture Notes in Computer Science. Springer, S. 113–130. DOI: 10.1007/978-3-540-76848-7_9.

Mendling, J., H. A. Reijers und J. Recker (2010). „Activity labeling in process modeling: Empirical insights and recommendations“. In: *Information Systems* 35.4, S. 467–482. DOI: 10.1016/j.is.2009.03.009.

Mendling, J., B. F. van Dongen und W. M. P. van der Aalst (2007b). „On the Degree of Behavioral Similarity between Business Process Models“. In: *6. GI Workshops Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten (EPK), St. Augustin, Deutschland*. Hrsg. von M. Nüttgens, F. J. Rump und A. Gadatsch. Bd. 303. CEUR Workshop Proceedings. CEUR, S. 39–58.

Mikolov, T., K. Chen, G. Corrado und J. Dean (2013a). „Efficient Estimation of Word Representations in Vector Space“. In: *Computing Research Repository (CoRR)* abs/1301.3781.

Mikolov, T., I. Sutskever, K. Chen, G. S. Corrado und J. Dean (2013b). „Distributed Representations of Words and Phrases and their Compositionality“. In: *27th Annual Conference on Neural Information Processing Systems, Lake Tahoe, Nevada, USA*. Hrsg. von C. J. C. Burges, L. Bottou, Z. Ghahramani und K. Q. Weinberger. Advances in Neural Information Processing Systems 26. Curran Associates, Inc., S. 3111–3119.

Miller, G. A. (1995). „WordNet: A Lexical Database for English“. In: *Communications of the ACM* 38.11, S. 39–41. DOI: 10.1145/219717.219748.

Minor, M., A. Tartakovski und R. Bergmann (2007). „Representation and Structure-Based Similarity Assessment for Agile Workflows“. In: *7th International Conference on Case-Based Reasoning (ICCBR), Belfast, Nordirland*. Hrsg. von R. Weber und M. M. Richter. Bd. 4626. Lecture Notes in Computer Science. Springer, S. 224–238. DOI: 10.1007/978-3-540-74141-1_16.

Momotko, M. und K. Subieta (2004). „Process Query Language: A Way to Make Workflow Processes More Flexible“. In: *8th East European Conference on Advances in Databases and Information Systems (ADBIS), Budapest, Ungarn*. Hrsg. von G. Gottlob, A. A. Benczúr und J. Demetrovics. Bd. 3255. Lecture Notes in Computer Science. Springer, S. 306–321. DOI: 10.1007/978-3-540-30204-9_21.

Montani, S., G. Leonardi, S. Quaglini, A. Cavallini und G. Micieli (2015). „A knowledge-intensive approach to process similarity calculation“. In: *Expert Systems with Applications* 42.9, S. 4207–4215. DOI: 10.1016/j.eswa.2015.01.027.

Müller, J. (2009). „A Rule-Based Approach to Match Structural Patterns with Business Process Models“. In: *International Symposium on Rule Interchange and Applications (RuleML), Las Vegas, USA*. Hrsg. von G. Governatori, J. Hall und A. Paschke. Bd. 5858. Lecture Notes in Computer Science. Springer, S. 208–215. DOI: 10.1007/978-3-642-04985-9_20.

Murata, T. (1989). „Petri nets: Properties, analysis and applications“. In: *Proceedings of the IEEE* 77.4, S. 541–580. DOI: 10.1109/5.24143.

Nakov, P., A. Popova und P. Mateev (2001). „Weight functions impact on LSA performance“. In: *Recent Advances in Natural Language Processing (RANLP), Tzigov Chark, Bulgarien*, S. 187–193.

Nejati, S., M. Sabetzadeh, M. Chechik, S. M. Easterbrook und P. Zave (2007). „Matching and Merging of Statecharts Specifications“. In: *29th International Conference on Software Engineering (ICSE), Minneapolis, USA*. IEEE Computer Society, S. 54–64. DOI: 10.1109/ICSE.2007.50.

Niedermann, F., S. Radeschütz und B. Mitschang (2010). „Design-Time Process Optimization through Optimization Patterns and Process Model Matching“. In: *12th IEEE Conference on Commerce and Enterprise Computing (CEC), Shanghai, China*. Hrsg. von K. Chao, C. Huemer, B. Hofreiter, Y. Li und N. Shah. IEEE Computer Society, S. 48–55. DOI: 10.1109/CEC.2010.9.

Niemann, M., M. Siebenhaar, J. Eckert und R. Steinmetz (2010). „Process Model Analysis Using Related Cluster Pairs“. In: *Business Process Management Workshops, Hoboken, USA*. Hrsg. von M. z. Muehlen und J. Su. Bd. 66. Lecture Notes in Business Information Processing. Springer, S. 547–558. DOI: 10.1007/978-3-642-20511-8.

Niemann, M., M. Siebenhaar, S. Schulte und R. Steinmetz (2012). „Comparison and retrieval of process models using related cluster pairs“. In: *Computers in Industry* 63.2, S. 168–180. DOI: 10.1016/j.compind.2011.11.002.

Niesen, T., S. Dadashnia, P. Fettke und P. Loos (2016). „A Vector Space Approach to Process Model Matching using Insights from Natural Language Processing“. In: *Multikonferenz Wirtschaftsinformatik (MKWI), Illmenau, Deutschland*. Hrsg. von V. Nissen, D. Stelzer, S. Straßburger und D. Fischer. Technische Universität Ilmenau, S. 93–104.

Norman, D. (1983). „Some Observations on Mental Models“. In: Hrsg. von D. Gentner und A. L. Stevens. Mental models. Lawrence Erlbaum Associates, Inc. Kap. 1, S. 7–14.

Oberweis, A. und V. Sänger (1994). „A graphical query language for simulation databases“. In: *Journal of Microcomputer Applications* 17.4, S. 345–367. DOI: 10.1006/jmca.1994.1024.

Oberweis, A. (1996). *Modellierung und Ausführung von Workflows mit Petri-Netzen*. Vieweg+Teubner, Wiesbaden. DOI: 10.1007/978-3-322-81039-7.

Pedersen, T., S. Patwardhan und J. Michelizzi (2004). „WordNet::Similarity – Measuring the Relatedness of Concepts“. In: *Demonstration Papers at HLT-NAACL 2004, Boston, USA*. Hrsg. von D. Palmer, J. Polifroni und D. Roy. Association for Computational Linguistics, S. 38–41. DOI: 10.3115/1614025.1614037.

Petri, C. A. (1962). „Kommunikation mit Automaten“. Diss. Universität Bonn.

Peukert, E., S. Maßmann und K. König (2010). „Comparing Similarity Combination Methods for Schema Matching“. In: *Informatik 2010, Leipzig, Deutschland*. Hrsg. von K.-P. Fähnrich und B. Franczyk. Bd. P-175. Lecture Notes in Informatics. Gesellschaft für Informatik, S. 692–701.

Pittke, F. (2016). *Linguistic Refactoring of Business Process Models*. Logos, Berlin.

Pittke, F., H. Leopold und J. Mendling (2013). „Spotting Terminology Deficiencies in Process Model Repositories“. In: *14th International Conference on Enterprise, Business-Process and Information Systems Modeling (BPMDS), Valencia, Spanien*. Hrsg. von S. Nurcan, H. A. Proper, P. Soffer, J. Krogstie, R. Schmidt, T. A. Halpin und I. Bider. Bd. 147. Lecture Notes in Business Information Processing. Springer, S. 292–307. DOI: 10.1007/978-3-642-38484-4_21.

Pittke, F., H. Leopold, J. Mendling und G. Tamm (2012). „Enabling Reuse of Process Models through the Detection of Similar Process Parts“. In: *Business Process Management Workshops, Tallinn, Estland*. Hrsg. von M. La Rosa und P. Soffer. Bd. 132. Lecture Notes in Business Information Processing. Springer, S. 586–597. DOI: 10.1007/978-3-642-36285-9_59.

Polyvyanyy, A., L. García-Bañuelos und M. Dumas (2012). „Structuring acyclic process models“. In: *Information Systems* 37.6, S. 518–538. DOI: 10.1016/j.is.2011.10.005.

Polyvyanyy, A., M. L. Rosa und A. H. M. ter Hofstede (2014). „Indexing and Efficient Instance-Based Retrieval of Process Models Using Untanglings“. In: *26th International Conference on Advanced Information Systems Engineering (CAiSE), Thessaloniki, Griechenland*. Hrsg. von M. Jarke, J. Mylopoulos, C. Quix, C. Rolland, Y. Manolopoulos, H. Mouratidis und J. Horkoff. Bd. 8484. Lecture Notes in Computer Science. Springer, S. 439–456. DOI: 10.1007/978-3-319-07881-6_30.

Polyvyanyy, A., A. H. M. ter Hofstede, M. La Rosa und C. Ouyang (2015). *Process Query Language: Design, Implementation and Evaluation*. BPM Center Report BPM-15-06. BPMcenter.org.

Porter, M. F. (1980). „An algorithm for suffix stripping“. In: *Program* 14.3, S. 130–137. DOI: 10.1108/eb046814.

Porter, M. E. (1985). *Competitive Advantage: Creating and Sustaining Superior Performance*. Free Press, New York, USA.

Prescher, J., C. D. Ciccio und J. Mendling (2014). „From declarative processes to imperative models“. In: *4th International Symposium on Data-Driven Process Discovery and Analysis (SIMPDA), Mailand, Italien*. Hrsg. von R. Accorsi, P. Ceravolo und B. Russo. Bd. 1293. CEUR Workshop Proceedings. CEUR, S. 162–173.

Qiao, M., R. Akkiraju und A. J. Rembert (2011). „Towards Efficient Business Process Clustering and Retrieval: Combining Language Modeling and Structure Matching“. In: *9th International Conference on Business Process Management (BPM), Clermont-Ferrand, Frankreich*. Hrsg. von S. Rinderle-Ma, F. Toumani und K. Wolf. Bd. 6896. Lecture Notes in Computer Science. Springer, S. 199–214. DOI: 10.1007/978-3-642-23059-2_17.

Rahm, E. und P. A. Bernstein (2001). „A Survey of Approaches to Automatic Schema Matching“. In: *The VLDB Journal* 10.4, S. 334–350. DOI: 10.1007/s007780100057.

Reisig, W. (1985). *Petri Nets – An Introduction*. Bd. 4. EATCS Monographs on Theoretical Computer Science. Springer, Berlin Heidelberg. DOI: 10.1007/978-3-642-69968-9.

Resnik, P. (1995). „Using Information Content to Evaluate Semantic Similarity in a Taxonomy“. In: *14th International Joint Conference on Artificial Intelligence (IJCAI), Montréal, Kanada*. Morgan Kaufmann Publishers, S. 448–453.

Rharbi, A., Z. Bakkoury, A. Betari und O. Boucelma (2010). *Reusability of Business Process Models Using a Querying Mechanism Based on OODBMS*.

Rinderle-Ma, S. und S. Kabicher-Fuchs (2016). „An Indexing Technique for Compliance Checking and Maintenance in Large Process and Rule Repositories“. In: *Enterprise Modelling and Information Systems Architectures* 11.2, S. 1–24. DOI: 10.18417/emisa.11.2.

Rodriguez, C., C. Klinkmüller, I. Weber, F. Daniel und F. Casati (2016). „Activity Matching with Human Intelligence“. In: *Business Process Management Forum (BPM Forum), Rio de Janeiro, Brasilien*. Hrsg. von M. L. Rosa, P. Loos und O. Pastor. Bd. 260. Lecture Notes in Business Information Processing. Springer, S. 124–140. DOI: 10.1007/978-3-319-45468-9_8.

Rus, V., M. C. Lintean, R. Banjade, N. B. Niraula und D. Stefanescu (2013). „SEMILAR: The Semantic Similarity Toolkit“. In: *51st Annual Meeting of the Association for Computational Linguistics, ACL, Conference System Demonstrations, Sofia, Bulgarien*. The Association for Computer Linguistics, S. 163–168.

Sakr, S., A. Awad und M. Kunze (2012). „Querying Process Models Repositories by Aggregated Graph Search“. In: *Business Process Management Workshops, Tallinn, Estland*. Hrsg. von M. L. Rosa und P. Soffer. Bd. 132. Lecture Notes in Business Information Processing. Springer, S. 573–585. DOI: 10.1007/978-3-642-36285-9_58.

Salton, G. und C. Buckley (1991). „Automatic Text Structuring and Retrieval: Experiments in Automatic Encyclopedia Searching“. In: *14th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Chicago, USA*. Hrsg. von A. Bookstein, Y. Chiaramella, G. Salton und V. V. Raghavan. ACM, S. 21–30. DOI: 10.1145/122860.122863.

Salton, G., A. Wong und C. S. Yang (1975). „A Vector Space Model for Automatic Indexing“. In: *Communications of the ACM* 18.11, S. 613–620. DOI: 10.1145/361219.361220.

Sánchez-Charles, D., V. Muntés-Mulero, J. Carmona und M. Solé (2016). „Process Model Comparison Based on Cophenetic Distance“. In: *Business Process Management Forum (BPM Forum), Rio de Janeiro, Brasilien*. Hrsg. von M. L. Rosa, P. Loos und O. Pastor. Bd. 260. Lecture Notes in Business Information Processing. Springer, S. 141–158. DOI: 10.1007/978-3-319-45468-9_9.

Sarno, R., H. Ginardi, E. W. Pamungkas und D. Sunaryono (2013). „Clustering of ERP business process fragments“. In: *International Conference on Computer, Control, Informatics and Its Applications (IC3INA), Jakarta, Indonesien*. IEEE Computer Society, S. 319–324. DOI: 10.1109/IC3INA.2013.6819194.

Schoknecht, A., N. Fischer und A. Oberweis (2017a). „Process Model Search Using Latent Semantic Analysis“. In: *Business Process Management Workshops: BPM 2016 International Workshops, Rio de Janeiro, Brasilien, Revised Papers*. Hrsg. von M. Dumas und M. Fantinato. Bd. 281. Lecture Notes in Business Information Processing. Springer, S. 283–295. DOI: 10.1007/978-3-319-58457-7_21.

Schoknecht, A. und A. Oberweis (2017). „LS3: Latent Semantic Analysis-based Similarity Search for Process Models“. In: *Enterprise Modelling and Information Systems Architectures* 12.2, S. 1–22. DOI: 10.18417/emisa.12.2.

Schoknecht, A., T. Thaler, P. Fettke, A. Oberweis und R. Laue (2017b). „Similarity of Business Process Models — A State-of-the-Art Analysis“. In: *ACM Computing Surveys* 50.4, 52:1–52:33. DOI: 10.1145/3092694.

Schuette, R. und T. Rotthowe (1998). „The Guidelines of Modeling – An Approach to Enhance the Quality in Information Models“. In: *17th International Conference on Conceptual Modeling (ER), Singapur*. Hrsg. von T.-W. Ling, S. Ram und M. Lee. Bd. 1507. Lecture Notes in Computer Science. Springer, S. 240–254. DOI: 10.1007/978-3-540-49524-6_20.

Shannon, C. E. (1948). „A mathematical theory of communication“. In: *The Bell System Technical Journal* 27.3, S. 379–423. DOI: 10.1109/9780470544242.ch1.

Shao, Q., P. Sun und Y. Chen (2009). „WISE: A Workflow Information Search Engine“. In: *25th International Conference on Data Engineering (ICDE), Shanghai, China*. Hrsg. von Y. E. Ioannidis, D. L. Lee und R. T. Ng. IEEE Computer Society, S. 1491–1494. DOI: 10.1109/ICDE.2009.89.

Skrinjar, R. und P. Trkman (2013). „Increasing process orientation with business process management: Critical practices“. In: *International Journal of Information Management* 33.1, S. 48–60. DOI: 10.1016/j.ijinfomgt.2012.05.011.

Sokolova, M. und G. Lapalme (2009). „A systematic analysis of performance measures for classification tasks“. In: *Information Processing and Management* 45.4, S. 427–437. DOI: 10.1016/j.ipm.2009.03.002.

SQL92: Database Language SQL (1992). Techn. Ber. ISO 9075.

Srivastava, B. und D. Mukherjee (2009). „Organizing Documented Processes“. In: *International Conference on Services Computing (SCC), Bangalore, Indien*. IEEE Computer Society, S. 25–32. DOI: 10.1109/SCC.2009.32.

Stachowiak, H. (1973). *Allgemeine Modelltheorie*. Springer, Wien, New York.

Stewart, G. W. (1993). „On the Early History of the Singular Value Decomposition“. In: *SIAM Review* 35.4, S. 551–566. DOI: 10.1137/1035134.

Störrle, H. und V. Acretoaie (2013). „Querying Business Process Models with VMQL“. In: *5th ACM SIGCHI Annual International Workshop on Behaviour Modelling - Foundations and Applications (BMFA), Montpellier, Frankreich*. ACM International Conference Proceeding Series. ACM, 4:1–4:10. DOI: 10.1145/2492437.2492441.

Sun, P. (2010). „Service Clustering Based on Profile and Process Similarity“. In: *3rd International Symposium on Information Science and Engineering (ISISE), Shanghai, China*. IEEE Computer Society, S. 535–539. DOI: 10.1109/ISISE.2010.151.

Taylor, F. W. (1911). *Principles of Scientific Management*. Harper, New York, USA.

Ter Hofstede, A. H. M., C. Ouyang, M. L. Rosa, L. Song, J. Wang und A. Polyvyanyy (2013). „APQL: A Process-Model Query Language“. In: *1st Asia Pacific Conference on Business Process Management (AP-BPM), Peking, China*. Hrsg. von M. Song, M. T. Wynn und J. Liu. Bd. 159. Lecture Notes in Business Information Processing. Springer, S. 23–38. DOI: 10.1007/978-3-319-02922-1_2.

Thaler, T., P. Hake, P. Fettke und P. Loos (2014). „Evaluating the Evaluation of Process Matching Techniques“. In: *Multikonferenz Wirtschaftsinformatik (MKWI), Paderborn, Deutschland*. Hrsg. von D. Kundisch, L. Suhl und L. Beckmann. Universität Paderborn, S. 1600–1612.

Thaler, T., A. Schoknecht, P. Fettke, A. Oberweis und R. Laue (2017). „A Comparative Analysis of Business Process Model Similarity Measures“. In: *Business Process Management Workshops: BPM 2016 International Workshops, Rio de Janeiro, Brasilien, Revised Papers*. Hrsg. von M. Dumas und M. Fantinato. Bd. 281. Lecture Notes in Business Information Processing. Springer, S. 310–322. DOI: 10.1007/978-3-319-58457-7_23.

Tka, M. und S. A. Ghannouchi (2012). „Comparison of Business Process Models as Part of BPR Projects“. In: *Procedia Technology* 5.0, S. 427–436. DOI: 10.1016/j.protcy.2012.09.047.

Turney, P. D. und P. Pantel (2010). „From Frequency to Meaning: Vector Space Models of Semantics“. In: *Journal of Artificial Intelligence Research* 37, S. 141–188. DOI: 10.1613/jair.2934.

Uba, R., M. Dumas, L. García-Bañuelos und M. La Rosa (2011). „Clone Detection in Repositories of Business Process Models“. In: *9th International Conference on Business Process Management (BPM), Clermont-Ferrand, Frankreich*. Hrsg. von S. Rinderle-Ma, F. Toumani und K. Wolf. Bd. 6896. Lecture Notes in Computer Science. Springer, S. 248–264. DOI: 10.1007/978-3-642-23059-2_20.

Van der Aa, H., A. Gal, H. Leopold, H. A. Reijers, T. Sagi und R. Shraga (2017). „Instance-Based Process Matching Using Event-Log Information“. In: *29th International Conference on Advanced Information Systems Engineering (CAiSE), Essen, Germany*. Hrsg. von E. Dubois und K. Pohl. Springer, S. 283–297. DOI: 10.1007/978-3-319-59536-8_18.

Van der Aalst, W. M. P. (1998). „The Application of Petri Nets to Workflow Management“. In: *Journal of Circuits, Systems, and Computers* 8.1, S. 21–66. DOI: 10.1142/S0218126698000043.

Van der Aalst, W. M. P. (2011). *Process Mining – Discovery, Conformance and Enhancement of Business Processes*. 1. Aufl. Springer, Berlin Heidelberg. DOI: 10.1007/978-3-642-19345-3.

Van der Aalst, W. M. P. (2013). „Business Process Management: A Comprehensive Survey“. In: *ISRN Software Engineering* 2013. DOI: 10.1155/2013/507984.

Van der Aalst, W. M. P., A. K. A. de Medeiros und A. J. M. M. Weijters (2006). „Process Equivalence: Comparing Two Process Models Based on Observed Behavior“. In: *4th International Conference on Business Process Management (BPM), Wien, Österreich*. Hrsg. von S. Dustdar, J. L. Fiadeiro und A. P. Sheth. Bd. 4102. Lecture Notes in Computer Science. Springer, S. 129–144. DOI: 10.1007/11841760_10.

Van Dongen, B. F., R. M. Dijkman und J. Mendling (2008). „Measuring Similarity between Business Process Models“. In: *20th International Conference on Advanced Information Systems Engineering (CAiSE), Montpellier, Frankreich*. Hrsg. von Z. Bellahsene und M. Léonard. Bd. 5074. Lecture Notes in Computer Science. Springer, S. 450–464. DOI: 10.1007/978-3-540-69534-9_34.

Van Dongen, S. und A. J. Enright (2012). „Metric distances derived from cosine similarity and Pearson and Spearman correlations“. In: *Computing Research Repository (CoRR)*.

Van Rijsbergen, C. J. (1979). *Information Retrieval*. 2. Aufl. Butterworth-Heinemann, Newton, MA, USA.

Vanhatalo, J., H. Völzer und F. Leymann (2007). „Faster and More Focused Control-Flow Analysis for Business Process Models Through SESE Decomposition“. In: *5th International Conference on Service-Oriented Computing (ICSOC), Wien, Österreich*. Hrsg. von B. J. Krämer, K. Lin und P. Narasimhan. Bd. 4749. Lecture Notes in Computer Science. Springer, S. 43–55. DOI: 10.1007/978-3-540-74974-5_4.

Vogelaar, J. J. C. L., H. M. W. Verbeek, B. Luka und W. M. P. van der Aalst (2011). „Comparing Business Processes to Determine the Feasibility of Configurable Models: A Case Study“. In: *Business Process Management Workshops, Clermont-Ferrand, Frankreich*. Hrsg. von F. Daniel, K. Barkaoui und S. Dustdar. Bd. 100. Lecture Notes in Business Information Processing. Springer, S. 50–61. DOI: 10.1007/978-3-642-28115-0_6.

Wang, J., T. He, L. Wen, N. Wu, A. H. M. ter Hofstede und S. Jianwen (2010). „A Behavioral Similarity Measure between Labeled Petri Nets Based on Principal Transition Sequences“. In: *On the Move to Meaningful Internet Systems (OTM), Confederated International Conferences CoopIS, IS, DOA and ODBASE, Part I, Hersonissos, Griechenland*. Hrsg. von R. Meersman, T. S. Dillon und P. Herrero. Bd. 6426. Lecture Notes in Computer Science. Springer, S. 394–401. DOI: 10.1007/978-3-642-16934-2.

Wang, J., S. Tan, L. Wen, R. K. Wong und Q. Guo (2012). „An empirical evaluation of process mining algorithms based on structural and behavioral similarities“. In: *Symposium on Applied Computing (SAC), Riva, Italien*. Hrsg. von S. Ossowski und P. Lecca. ACM, S. 211–213. DOI: 10.1145/2245276.2245316.

Wang, Y., M. Li, J. Cao, X. Lin und F. Tang (2007). „Workflow Similarity Measure for Process Clustering in Grid“. In: *4th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD), Haikou, China*. Hrsg. von J. Lei. IEEE Computer Society, S. 629–635. DOI: 10.1109/FSKD.2007.618.

Wasser, A. und M. Lincoln (2013). „Selection of Business Process Alternatives Based on Operational Similarity for Multi-subsidiary Organizations“. In: *On the Move to Meaningful Internet Systems (OTM), Confederated International Workshops: OTM Academy, OTM Industry Case Studies Program (OTM Workshops), Graz, Austria*. Hrsg. von Y. T. Demey und H. Panetto. Bd. 8186. Lecture Notes in Computer Science. Springer, S. 579–586. DOI: 10.1007/978-3-642-41033-8_73.

Wasser, A., M. Lincoln und R. Karni (2006). „ProcessGene Query – A Tool for Querying the Content Layer of Business Process Models“. In: *BPM Demo Session at the 4th International Conference on Business Process Management (BPM), Wien, Österreich*. Hrsg. von J. Mendling. Bd. 203. CEUR Workshop Proceedings. CEUR.

Weber, M. und E. Kindler (2003). „The Petri Net Markup Language“. In: *Petri Net Technology for Communication-Based Systems – Advances in Petri Nets*. Hrsg. von H. Ehrig, W. Reisig, G. Rozenberg und H. Weber. Bd. 2472. Lecture Notes in Computer Science. Springer, S. 124–144. DOI: 10.1007/978-3-540-40022-6_7.

Webster, J. und R. T. Watson (2002). „Analyzing the Past to Prepare for the Future: Writing a Literature Review“. In: *MIS Quarterly* 26.2, S. xiii–xxiii.

Weidlich, M., R. Dijkman und J. Mendling (2010). „The ICoP Framework: Identification of Correspondences between Process Models“. In: *22nd International Conference on Advanced Information Systems Engineering (CAiSE), Hammamet, Tunesien*. Hrsg. von B. Pernici. Bd. 6051. Lecture Notes in Computer Science. Springer, S. 483–498. DOI: 10.1007/978-3-642-13094-6_37.

Weidlich, M., J. Mendling und M. Weske (2011). „Efficient Consistency Measurement Based on Behavioral Profiles of Process Models“. In: *IEEE Transactions on Software Engineering* 37.3, S. 410–429. DOI: 10.1109/TSE.2010.96.

Weidlich, M., E. Sheetrit, M. C. Branco und A. Gal (2013). „Matching Business Process Models Using Positional Passage-Based Language Models“. In: *32nd International Conference on Conceptual Modeling (ER), Hong-Kong, China*. Hrsg. von W. Ng, V. C. Storey und J. Trujillo. Bd. 8217. Lecture Notes in Computer Science. Springer, S. 130–137. DOI: 10.1007/978-3-642-41924-9.

Weske, M. (2012). *Business Process Management – Concepts, Languages, Architectures*. 2. Aufl. Springer, Berlin Heidelberg. DOI: 10.1007/978-3-642-28616-2.

Wilmont, I., S. Hengeveld, E. Barendsen und S. Hoppenbrouwers (2013). „Cognitive Mechanisms of Conceptual Modelling“. In: *32nd International Conference on Conceptual Modeling (ER), Hong-Kong, China*. Hrsg. von W. Ng, V. Storey und J. Trujillo. Bd. 8217. Lecture Notes in Computer Science. Springer, S. 74–87. DOI: 10.1007/978-3-642-41924-9_7.

Wombacher, A. (2006). „Evaluation of Technical Measures for Workflow Similarity Based on a Pilot Study“. In: *On the Move to Meaningful Internet Systems (OTM), Confederated International Conferences CoopIS, DOA, GADA, and ODBASE, Part I, Montpellier, Frankreich*. Hrsg. von R. Meersman und Z. Tari. Bd. 4275. Lecture Notes in Computer Science. Springer, S. 255–272. DOI: 10.1007/11914853_16.

Wombacher, A., P. Fankhauser, B. Mahleko und E. J. Neuhold (2004). „Matchmaking for Business Processes Based on Choreographies“. In: *International Journal of Web Services Research* 1.4, S. 14–32. DOI: 10.4018/jwsr.2004100102.

Wombacher, A. und C. Li (2010). „Alternative Approaches for Workflow Similarity“. In: *International Conference on Services Computing (SCC), Miami, USA*. IEEE Computer Society, S. 337–345. DOI: 10.1109/SCC.2010.95.

Wombacher, A. und M. Rozie (2006). „Evaluation of Workflow Similarity Measures in Service Discovery“. In: *Service Oriented Electronic Commerce, Passau, Deutschland*. Hrsg. von M. Schoop, C. Huemer, M. Rebstock und M. Bichler. Bd. P-80. Lecture Notes in Informatics. Gesellschaft für Informatik, S. 51–71.

Wu, Z. und M. S. Palmer (1994). „Verb Semantics and Lexical Selection“. In: *32nd Annual Meeting of the Association for Computational Linguistics, Las Cruces, USA*. Hrsg. von J. Pustejovsky. Morgan Kaufmann Publishers / ACL, S. 133–138.

Xiao, L., L. Zheng, J. Xiao und Y. Huang (2009). „A Graphical Query Language for Querying Petri Nets“. In: *3rd International United Information Systems Conference on Information Systems: Modeling, Development, and Integration (UNISCON), Sydney, Australien*. Hrsg. von J. Yang, A. Ginige, H. C. Mayr und R. Kutsche. Bd. 20. Lecture Notes in Business Information Processing. Springer, S. 514–525. DOI: 10.1007/978-3-642-01112-2_52.

Yan, Z., R. M. Dijkman und P. W. P. J. Grefen (2012a). „Business process model repositories – Framework and survey“. In: *Information & Software Technology* 54.4, S. 380–395. DOI: 10.1016/j.infsof.2011.11.005.

Yan, Z., R. M. Dijkman und P. W. P. J. Grefen (2012b). „FNet: An Index for Advanced Business Process Querying“. In: *10th International Conference on Business Process Management (BPM), Tallinn, Estland*. Hrsg. von A. P. Barros, A. Gal und E. Kindler. Bd. 7481. Lecture Notes in Computer Science. Springer, S. 246–261. DOI: 10.1007/978-3-642-32885-5_20.

Yan, Z., R. Dijkman und P. Grefen (2010). „Fast Business Process Similarity Search with Feature-Based Similarity Estimation“. In: *On the Move to Meaningful Internet Systems (OTM), Confederated International Conferences CoopIS, IS, DOA and ODBASE, Part I, Hersonissos, Griechenland*. Hrsg. von R. Meersman, T. Dillon und P. Herrero. Bd. 6426. Lecture Notes in Computer Science. Springer, S. 60–77. DOI: 10.1007/978-3-642-16934-2_8.

Yan, Z., R. Dijkman und P. Grefen (2012c). „Fast business process similarity search“. In: *Distributed and Parallel Databases* 30.2, S. 105–144. DOI: 10.1007/s10619-012-7089-z.

Zaman, A. N. K. und C. G. Brown (2010). „Latent Semantic Indexing and Large Dataset: Study of Term-Weighting Schemes“. In: *5th International Conference on Digital Information Management (ICDIM), Thunder Bay, USA*. IEEE Computer Society, S. 1–4. DOI: 10.1109/ICDIM.2010.5664669.

Zezula, P., G. Amato, V. Dohnal und M. Batko (2006). *Similarity Search – The Metric Space Approach*. Bd. 32. Advances in Database Systems. Springer, Philadelphia, New York, USA. DOI: 10.1007/0-387-29151-2.

Zha, H., J. Wang, L. Wen und C. Wang (2009). „A Label-Free Similarity Measure between Workflow Nets“. In: *4th Asia-Pacific Services Computing Conference (APSCC), Singapur*. Hrsg. von M. Kirchberg, P. C. K. Hung, B. Carminati, C. Chi-Hung, R. Kanagasabai, E. D. Valle, K. Lan und L. Chen. IEEE Computer Society, S. 463–469. DOI: 10.1109/APSCC.2009.5394086.

Zha, H., J. Wang, L. Wen, C. Wang und J. Sun (2010). „A Workflow Net Similarity Measure Based on Transition Adjacency Relations“. In: *Computers in Industry* 61.5, S. 463–471. DOI: 10.1016/j.compind.2010.01.001.

Zhuge, H. (2002). „A process matching approach for flexible workflow process reuse“. In: *Information and Software Technology* 44.8, S. 445–450. DOI: 10.1016/S0950-5849(02)00022-8.

Zloof, M. M. (1977). „Query-by-example: A Data Base Language“. In: *IBM Systems Journal* 16.4, S. 324–343. DOI: 10.1147/sj.164.0324.

Zloof, M. M. (1982). „Office-by-example: A Business Language That Unifies Data and Word Processing and Electronic Mail“. In: *IBM Systems Journal* 21.3, S. 272–304. DOI: 10.1147/sj.213.0272.

Ähnlichkeitsbasierte Suche in Geschäftsprozessmodelldatenbanken

Unternehmen wenden sich zunehmend einer prozessorientierten Sichtweise zu. Dabei werden Prozessmodelle dazu verwendet Prozesse zu dokumentieren, zu analysieren und zu verbessern. Die Erstellung von Geschäftsprozessmodellen erfolgt typischerweise durch ein iteratives Vorgehen, das einen hohen Zeitaufwand sowie hohe Kosten verursacht. Die Wiederverwendung existierender Modelle bietet sich zur Reduzierung dieser Nachteile an. Allerdings ist das Auffinden von zu einem neu zu modellierenden Geschäftsprozess ähnlichen Modellen durch große Modellsammlungen manuell nicht effizient möglich. Hilfreich sind in diesem Fall Suchmöglichkeiten nach relevanten Modellen, die als Vorlage zur Modellierung genutzt werden können. In dieser Arbeit werden verschiedene Ansätze beschrieben, um innerhalb von Prozessmodellbibliotheken zu suchen, sodass ein Modellierer existierende Modelle wiederverwenden kann.

ISBN 978-3-7315-0812-0