

Reader zu Graphen in den Digitalen Geisteswissenschaften

Modellierung – Import – Analyse

Andreas Kuczera

Inhaltsverzeichnis

1 Einleitung	4
1.1 Warum ein Reader zu Graphentechnologien in den digitalen Geisteswissenschaften?	4
1.2 Zum Aufbau des Readers	4
2 Einführung und theoretische Grundlagen	6
2.1 Einführung zu Graphen	6
2.2 Herkunft und Idee	7
2.3 Graphtypen und Strukturen	8
2.3.1 Graph und nicht Graph	8
2.3.2 Labeled Property Graph	8
2.3.3 Einfache Graphtypen	9
2.3.4 Verbundene und nichtverbundene Graphen	11
2.3.5 Gewichtete und ungewichtete Graphen	14
2.3.6 Gerichtete und ungerichtete Graphen	14
2.4 Installation und Start	16
2.5 Zusammenfassung	17
3 Das Projekt Regesta Imperii oder “Wie suchen Onlinenutzer Regesten?”	18
3.1 Das Projekt Regesta Imperii	18
3.2 Die Digitalisierung der Regesta Imperii	20
3.3 Wie suchen Online-Nutzer Regesten ?	20
3.4 Historische Netzwerkanalyse in den Registern	22
3.5 Zusammenfassung	24
4 Regestenmodellierung im Graphen	27
4.1 Wie kommen die Regesten in den Graphen	27
4.1.1 Import mit dem LOAD CSV-Befehl	28
4.1.2 Regestenmodellierung im Graphen	28
4.1.3 Indexe Erstellen	30
4.1.4 Erstellen der Regestenknoten	30
4.1.5 Erstellen der Ausstellungsorte	31
4.1.6 Koordinaten der Ausstellungsorte	33
4.1.7 Ausstellungsdatum	33
4.2 Exkurs 1: Herrscherhandeln in den Regesta Imperii	34
4.3 Zitationsnetzwerke in den Regesta Imperii	34

4.4	Import der Registerdaten in die Graphdatenbank	35
4.4.1	Vorbereitung der Registerdaten	35
4.4.2	Import der Registerdaten in die Graphdatenbank	37
4.5	Exkurs 2: Die Hierarchie des Registers der Regesten Kaiser Friedrichs III.	38
4.6	Auswertungsperspektiven	39
4.6.1	Personennetzwerke in den Registern	39
4.6.2	Herrschерhandeln ausgezählt	47
4.6.3	Herrschерhandeln pro Ausstellungsort ausgezählt	48
4.6.4	Herrschерhandeln und Anwesenheit	49
4.6.5	Regesten 200 km rund um Augsburg	50
4.6.6	Welche Literatur wird am meisten zitiert	50
4.6.7	Der Import zusammengefasst	51
4.7	Zusammenfassung	51
5	Verwandtschaft im Graphen	52
5.1	Das Projekt Nomen et Gens	52
5.2	Nomen et Gens im Graphen	52
5.3	Sind Berchar und Karl der Große verwandt ?	54
5.4	Zusammenfassung	54
6	Zusammenfassung	56

1 Einleitung

1.1 Warum ein Reader zu Graphentechnologien in den digitalen Geisteswissenschaften?

Graphentechnologien sind hervorragend für die Modellierung, Speicherung und Analyse hochvernetzter Daten geeignet. Obgleich als Konzept schon länger etabliert, erlebten sie mit dem Aufkommen des Internets und der Social-Media-Welle einen Aufschwung. Gegenüber relationalen Datenbankmodellen, bei denen die Daten in miteinander verknüpften Tabellen gespeichert werden, sind die Informationen in Graphdatenbanken in Knoten und Kanten modelliert und auf Specherebene auch genau so abgelegt. Mit diesem, einer Mind-Map sehr ähnlichen Modell lassen sich Forschungsdaten und Forschungsfragestellungen in einer Weise modellieren, die dem menschlichen Denken oft sehr nahe kommt.

Gerade in den digitalen Geisteswissenschaften gelingt es mit dem Graphenmodell bei der Modellierung und Strukturierung von Forschungsdaten und Forschungsfragestellung die Kluft zwischen Informatiker und Geisteswissenschaftler zu schließen, da der Graph eine für beide Seiten verständliche Plattform bietet. Für den Informatiker ist er hinreichend genau und berechenbar und für den Geisteswissenschaftler wegen seiner Schema- und Hierarchiefreiheit ausreichend flexibel. Gerade diese Eigenschaften, mit denen sich die beiden zentralen Zweige der Digitalen Geisteswissenschaften vereinen lassen, machen Graphen zu einem Schlüsselkonzept der Geisteswissenschaften des 21 Jahrhunderts.

1.2 Zum Aufbau des Readers

Der vorliegende Reader ist als einführende Lektüre konzipiert. Zu Beginn wird anhand einfacher Beispiele in die Grundlagen der Graphentechnologie eingeführt und die Verwendung von Graphdatenbanken erklärt.

Im nächsten Abschnitt werden Beispiele für die Modellierung und den Import bereits vorhandener Forschungsdaten aus den Projekten Nomen-et-Gens und den Regesta Imperii in eine Graphdatenbank vorgestellt.

Parallel zu den Kapiteln werden interaktive Übungen, sogenannte Graphgists angeboten, in denen die Studierenden ihr Wissen testen und überprüfen können.

Mit der Vermittlung der Konzepte von Graphen und Graphdatenbanken werden Kompetenzen in den Bereichen Modellierung (was hängt wie zusammen), Quellenkritik (welche

Qualität haben die Informationen in Knoten und Kanten) und der Verknüpfung von wissenschaftlicher Fragestellung und ihrer digitalen Modellierung geschärft. Daher vermittelt dieser Reader dem Leser wichtiges Rüstzeug für die zunehmende Digitalisierung immer größerer, auch die Geisteswissenschaften umfassender, Bereiche der Gesellschaft.

Für Fragen und Rückmeldungen stehe ich gerne zur Verfügung und wünsche beim Studium viel Freude.

Gießen, im August 2019

Andreas Kuczera

2 Einführung und theoretische Grundlagen

2.1 Einführung zu Graphen

Dieser Abschnitt gibt eine kurze Einführung in Graphen.¹ Gegenüber relationalen Datenbanken, in denen die Daten in Tabellen abgelegt und verknüpft sind, werden in Graphen Knoten für die Speicherung der Daten verwendet und diese dann mit Kanten in Relation gesetzt. Aber auch in relationalen Datenbanken kann man natürlich Daten verknüpfen. Wo liegt also der eigentliche Vorteil?

Der Beispielgraph in Abb. 2.1 modelliert einen Zusammenhang aus dem Frühmittelalter und erklärt kurz die Grundprinzipien in der geisteswissenschaftlichen Domäne.

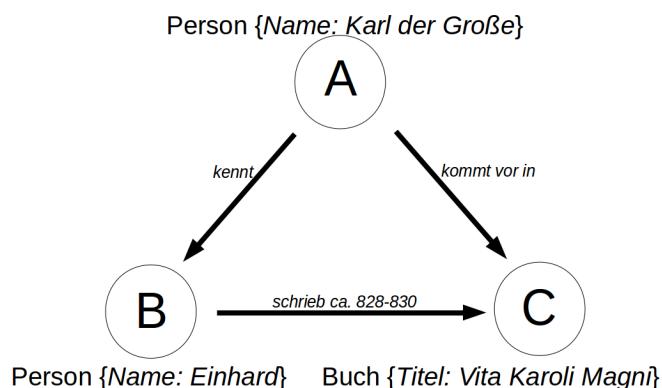


Abbildung 2.1: Beispielgraph (Quelle: Kuczera)

Der Beispielgraph zeigt oben einen Knoten (engl. Node) vom Typ (engl. Label) Person mit der Eigenschaft (engl. Property) Name. Diese hat den Wert “Karl der Große”. Links unten ist ein weiter Knoten vom Typ Person mit dem Namen “Einhard”. Rechts unten ist ein Knoten vom Typ Buch und dem Titel “Vita Karoli Magni” abgebildet. Die Kanten

¹Dieser Abschnitt beruht in Teilen auf den Kapiteln “Introduction” und “Graph Theory and Concepts” des Buches Graph Algorithms: Practical Examples in Apache Spark and Neo4j von Mark Needham und Amy E. Hodler, O’Reilly Media 2019 (<https://www.oreilly.com/library/view/graph-algorithms/9781492047674/>), S. xiii-xv und S. 1-26.

(engl. Edges) geben an, dass Karl der Große Einhard kannte, Einhard ca. 828-830 das Buch “Vita Karoli Magni” schrieb und Karl der Große in dem Buch vorkommt.

Knoten und Kanten können also noch zusätzliche Eigenschaften besitzen, in denen weitere Informationen gespeichert sind. Diese Eigenschaften sind spezifisch für die jeweiligen Knotentypen. So sieht man in der Abbildung, dass die beiden Knoten vom Typ Person jeweils noch die Eigenschaft Namen haben, deren Wert dann die Namen der Person angibt, während der Knoten vom Typ Buch die Eigenschaft Titel trägt, in dem der Titel des Buches abgespeichert wird.

Der wirkliche Mehrwert bei Graphdatenbanken ergibt sich aus gerichteten (also Verbindungen mit einer Richtung) und transitiven Beziehungen. Hat A eine direkte Kante zu B und B eine direkte Kante zu C dann ist A nicht direkt sondern transitiv mit C verbunden. Gerade wenn ein Graph sehr viele verschiedene solcher transitiven Verbindungen hat, lassen sich Muster und Verbindungen identifizieren, die in relationalen Modellen oft unentdeckt bleiben. Darüber hinaus bietet der Graph eine optimale Ausgangslage für anschließende Netzwerkanalyse des gesamten Graphen oder ausgewählter Subgraphen. War es in den digitalen Geisteswissenschaften bis vor einigen Jahren noch höchste Priorität überhaupt digitale Forschungsdaten bereitzustellen ist es heute die Herausforderung Daten in ihrem Kontext zu erfassen. Hierfür lassen sich Graphdatenbanken hervorragend nutzen.

2.2 Herkunft und Idee

Graphen gehen zurück auf die erste Hälfte des 18 Jahrhunderts. 1736 löste Leonhard Euler das “Königsberger Brückenproblem”, das fragte ob es möglich sei, die 4 durch den Fluss getrennten Stadtbereiche, die über 7 Brücken verbunden waren jeweils einmal zu besuchen, ohne eine der Brücken zweimal zu nutzen (Abb. 2.2).²

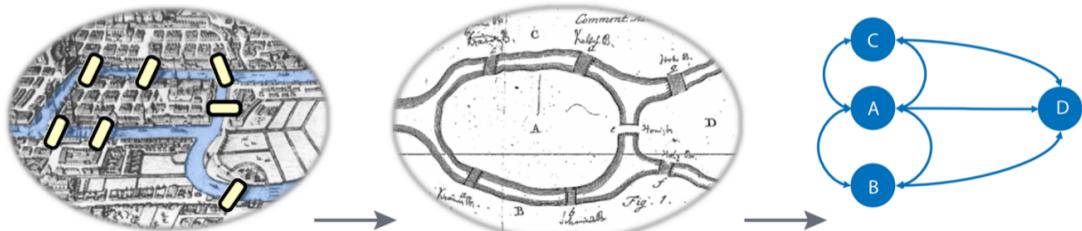


Abbildung 2.2: Das Eulersche Brückenproblem (Needham/Hodler 2019, S. 2, Bild 1-1)

Euler kam zu der Einsicht, dass nur die Verbindungen zwischen den 4 Bereichen der Stadt relevant sind und legte damit den Grundstein für die Graphtheorie und ihre Mathematik. Er zeigte, dass das Brückenproblem nicht lösbar war, da zu jedem Stadtbereich eine

²Vgl. https://de.wikipedia.org/wiki/K%C3%B6nigsberger_Br%C3%BCckenproblem.

ungerade Anzahl von Brücken führte, es dürfe aber nur zwei Ufer mit einer ungeraden Anzahl geben.³

2.3 Graphtypen und Strukturen

2.3.1 Graph und nicht Graph

Auch wenn Graphen ihren Ursprung in der Mathematik haben so sind sie doch ein pragmatisches Werkzeug um Informationen zu modellieren und zu analysieren.

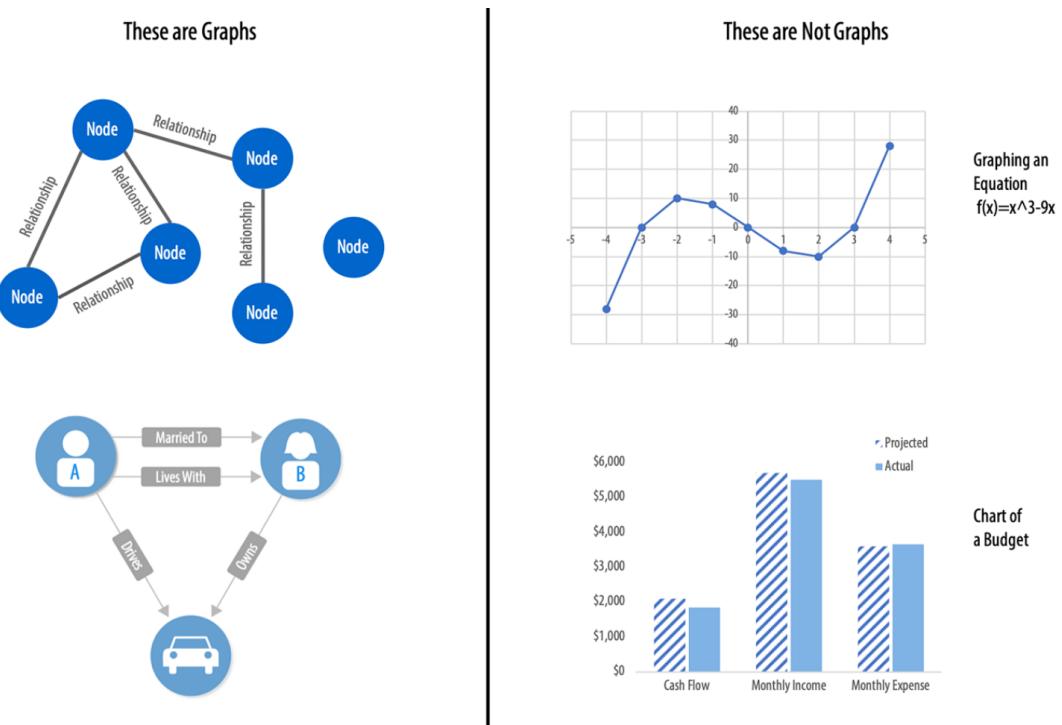


Abbildung 2.3: Graphen und nicht Graphen (Needham/Hodler 2019, S. 3, Bild 1-2)

In diesem Reader geht es nicht um Graphen im Sinne von Lösung von Gleichungen wie in Abb. 2.3 rechts. Es geht um Informationen, die durch Knoten und Kanten dargestellt werden, wie links im Bild.

2.3.2 Labeled Property Graph

Einen Graphen, in dem die Knoten und Kanten Typen (Labels) und Eigenschaften (Properties) besitzen, nennt man “Labeled Property Graph” (LPG).

³Vgl. https://de.wikipedia.org/wiki/K%C3%B6nigsberger_Br%C3%BCckenproblem.

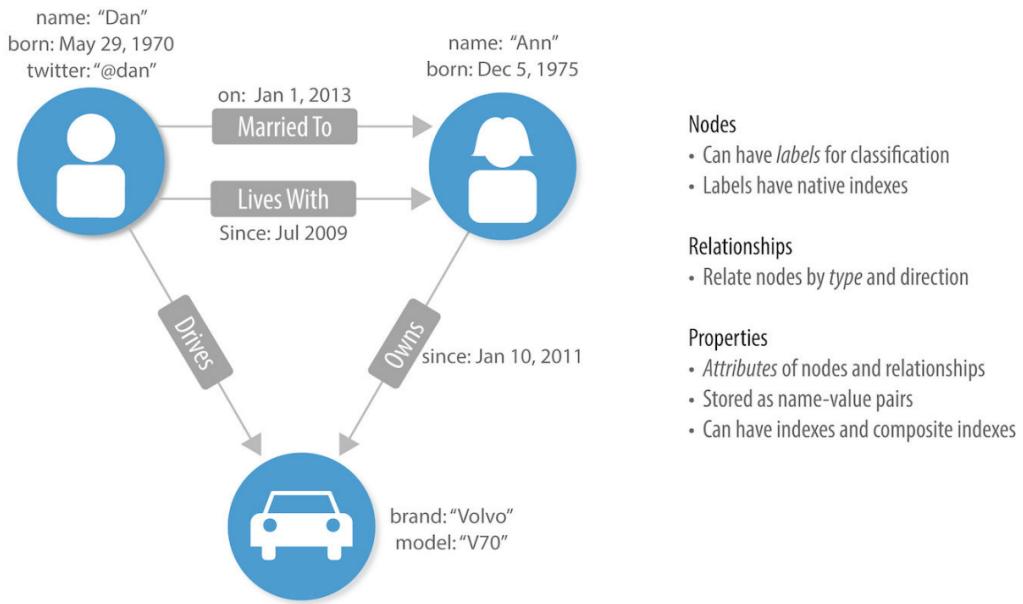


Abbildung 2.4: Beispiel für einen Labeled-Property-Graphen (Needham/Hodler 2019, S. 16, Bild 2-1)

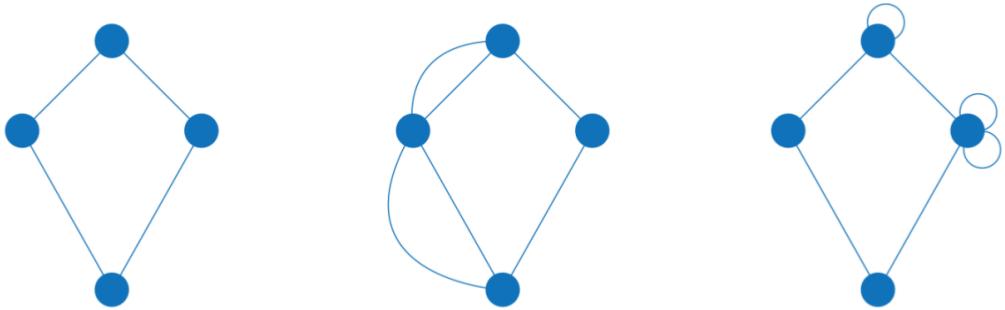
Der Graph in Abb. 2.4 sagt beispielsweise aus, dass der Knoten links oben das Label “männliche Person” mit den zugehörigen Properties (name:’Dan’, born etc.) trägt. Rechts daneben ist eine weibliche Person mit ihren Properties (name:’Ann’, born: Dec 5, 1975) abgebildet. Die Kanten sagen aus, dass Ann am 1. Januar 2013 mit Dan zusammenlebte und seit Juli 2009 mit ihm verheiratet ist . Ann gehört gleichzeitig ein Auto (mit der Property Marke: ‘Volvo’), das von Dan gefahren wird. Hier werden in einfachen Schritten Informationen aus der “realen Welt” abgebildet. In den nächsten Abschnitten geht es nun um die Konzepte dahinter.

2.3.3 Einfache Graphtypen

In Abb. 2.5 sind verschiedene Graphtypen abgebildet. Im Beispiel links ist ein einfacher Graph dargestellt, in dem jedes Knotenpaar nur eine Verbindung haben kann.

Im mittleren Beispiel sind mehrere Verbindungen zwischen Knotenpaaren möglich. Das rechte Beispiel ergänzt sich noch um die Möglichkeit, Verbindungen von Knoten wieder zu ihnen selbst zurück zu ermöglichen.

Die untere Bildhälfte zeigt links ein Beispiel für einen zufälligen Graphen, aus dem sich keine hierarchischen oder strukturellen Informationen ablesen lassen.

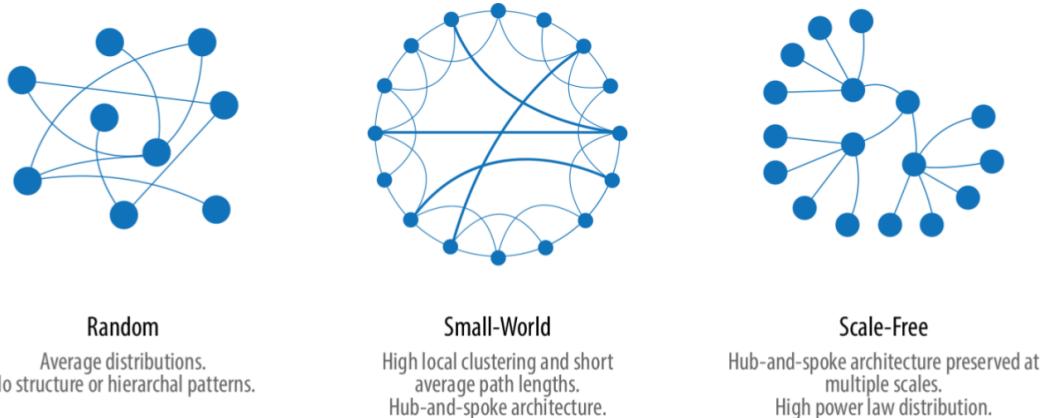


Simple Graph
Node pairs can only have one relationship between them.

Multigraph
Node pairs can have multiple relationships between them.

Graph (also Pseudograph)
Node pairs can have multiple relationships between them.
Nodes can loop back to themselves.

Abbildung 2.5: Einfache Graphtypen (Needham/Hodler 2019, S. 17, Bild 2-2)



In der Mitte ist ein Small-World-Graph abgebildet, der lokal stark geclustert ist und eine geringe durchschnittliche Pfadlänge hat. Die rechte Abbildung zeigt ein skalenfreies Netzwerk, wie beispielsweise das World-Wide-Web.

In Tab. 2.1 werden charakteristische Möglichkeiten zur Typisierung von Graphen gegenübergestellt.

Graphattribute	Unterschiede	Überlegungen zum Algorithms
Verbundene und nichtverbundene Graphen	Gibt es einen Pfad (Länge egal) zwischen zwei beliebigen Knoten eines Graphen	Knoteninseln können zu unvorhergesehenem Verhalten, wie Abbruch oder Auslassen von Knoteninseln führen

Graphattribute	Unterschiede	Überlegungen zum Algorithms
Gewichtete und ungewichtete Graphen	Gibt es (domänen-spezifische) Werte für Wichtungen an Knoten und Kanten	Werte machen den Graphen reicher an Informationen; Viele Algorithmen benötigen gewichtende Werte
Gerichtete und ungerichtete Graphen	Ist eine Verbindung hierarchisch oder gegenseitig	Die Richtung einer Kante sind für viele Auswertungen von Relevanz
Zyklische und nicht zyklische Graphen	In zyklischen Graphen können Pfade können wieder zum Startknoten zurückkehren	Zyklische Graphen sind weit verbreitet. Bei der Auswertung mit Graphalgorithmen muss der Graph aber möglicherweise bearbeitet werden
Graphdichte	Das Verhältnis von Kanten zu Knotenzahl	Extrem dichte oder extrem dünne Graphen können die Analyse erschweren. Ggf. können Änderungen an der Modellierung helfen, sofern es die Domäne zulässt
Monopartite, bipartite, und k-partite Graphen	Gibt es einen, zwei oder mehrere Knotentypen	Mehr Knotentypen erleichtern in den Geisteswissenschaften die Modellierung, Graphalgorithmen arbeiten oft aber nur mit einem Knotentyp

Quelle: <https://www.oreilly.com/library/view/graph-algorithms/9781492047674/>, S. 18-19.

2.3.4 Verbundene und nichtverbundene Graphen

Sind in einem Graphen mehrere Gruppen von verbundenen Knoten vorhanden und es gibt zwischen den Gruppen keine Verbindungen, handelt es sich um nichtverbundene Graphen.

Das Beispiel in Abb. 2.6 zeigt einen Graphen mit Personen und ihren Verwandtschaftsbeziehungen. Es gibt mehrere Stammbäume, die nicht miteinander verbunden sind.

Abb. 2.7 zeigt einen verbundenen Graphen, bei dem alle Knoten mindestens eine Kante haben und jeder Knoten jeden anderen über einen Pfad erreichen kann. Hier wird ein Ausschnitt aus einer Graphdatenbank gezeigt, in die eine XML-Text-Datei als XML-Baum importiert wurde. Da es sich bei jeder XML-Datei um einen Baum handelt und jeder XML-Knoten Teil dieses Baumes ist, gibt es keine nichtverbundenen Teile.

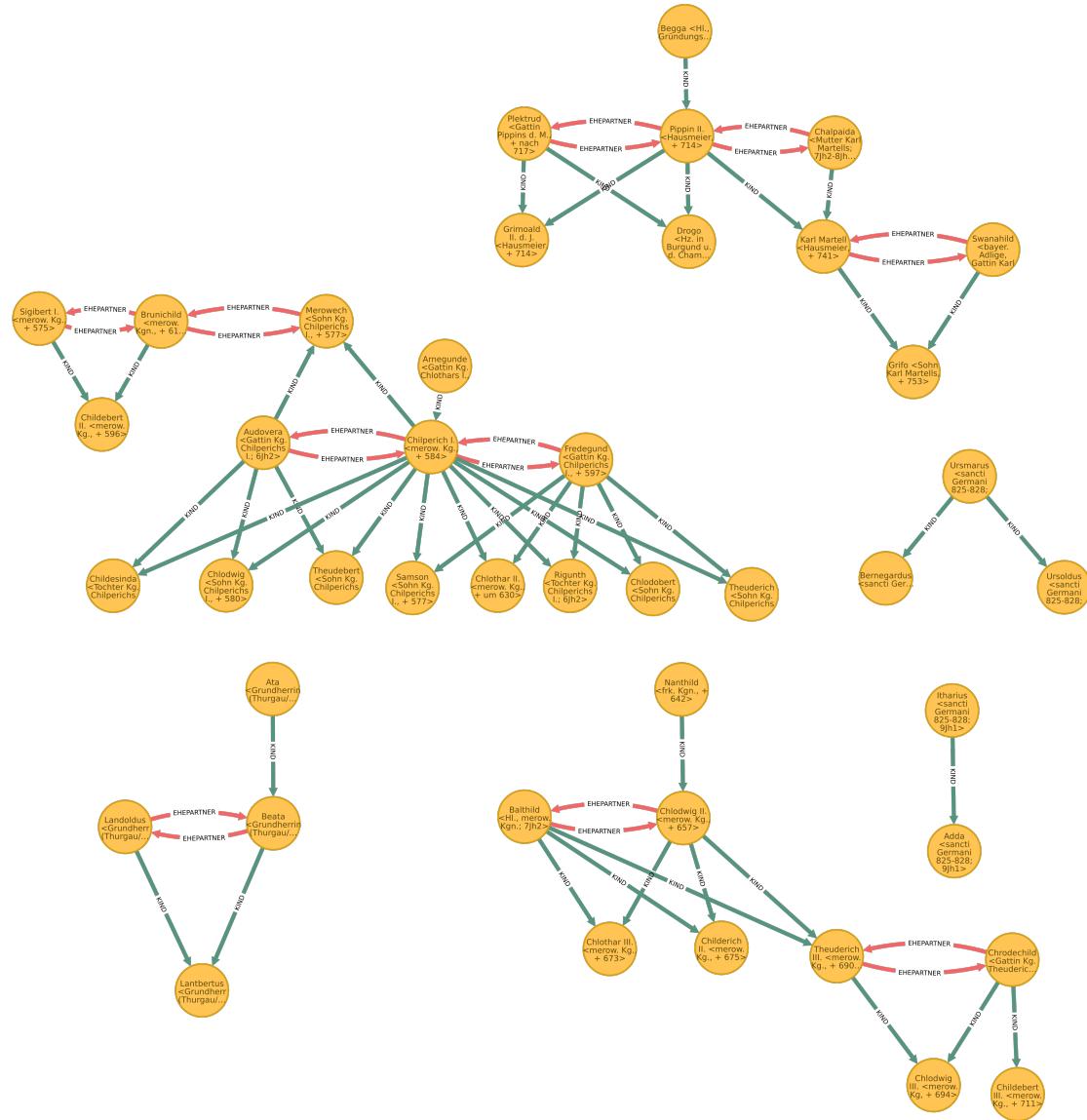


Abbildung 2.6: Nichtverbundener Graph (Quelle: Kuczera)

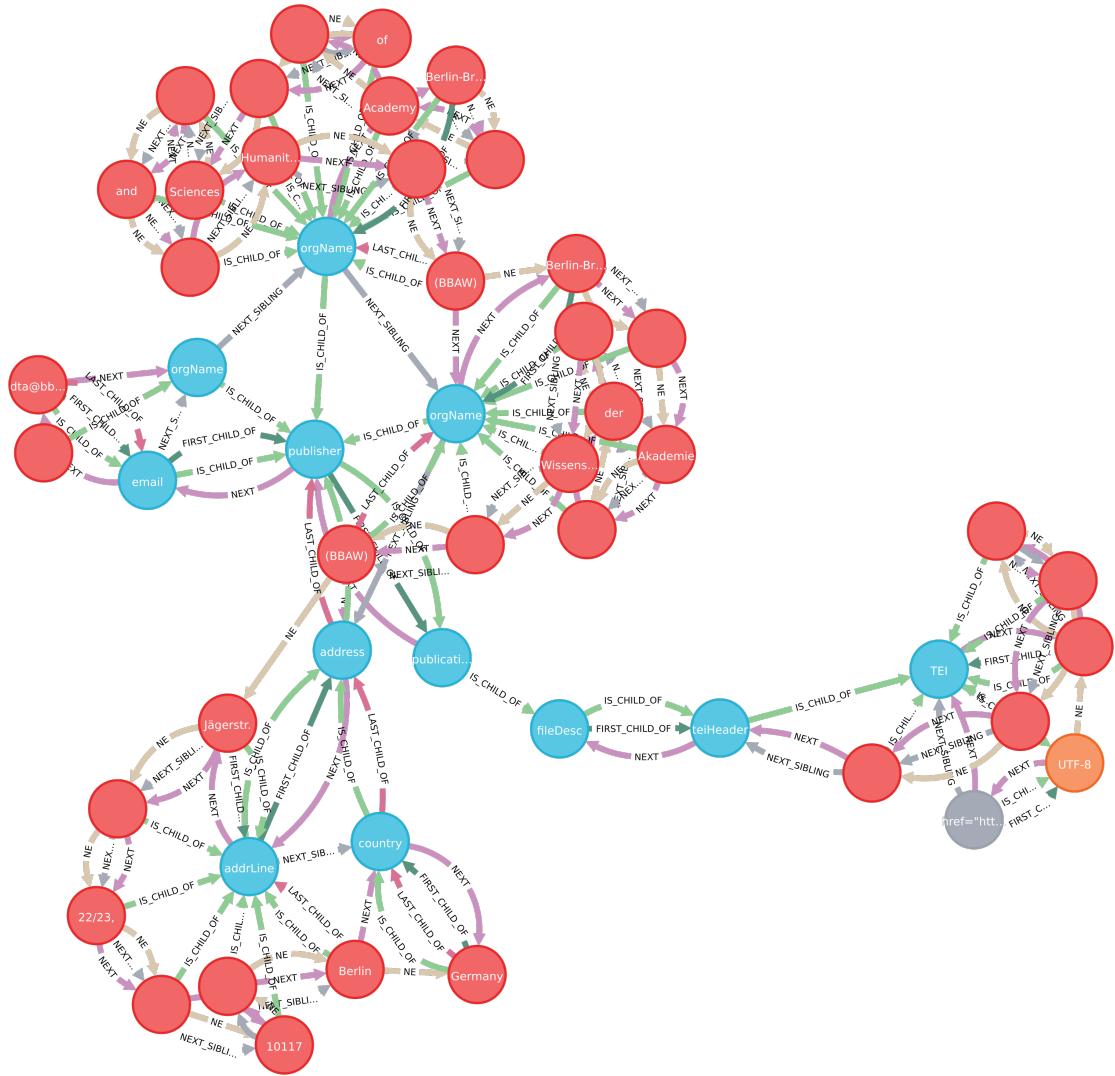


Abbildung 2.7: Verbundener Graph (Quelle: Kuczera)

2.3.5 Gewichtete und ungewichtete Graphen

In ungewichteten Graphen besitzen die Kanten keinen Wert zu Gewichtung.

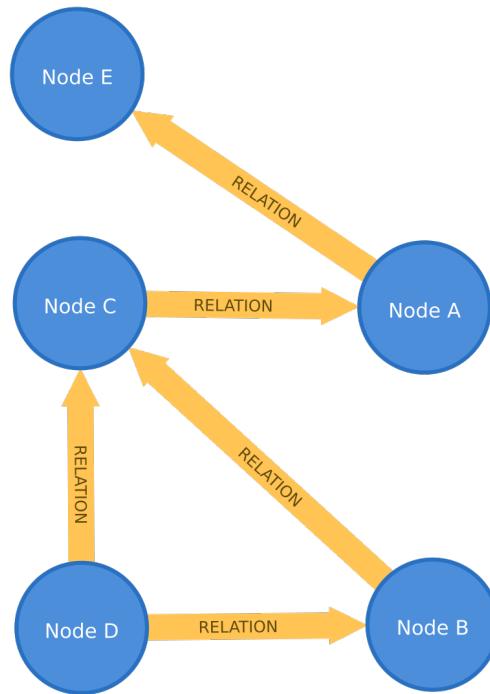


Abbildung 2.8: Ungewichteter Graph (Quelle: Kuczera)

Im Graphen in Abb. 2.8 werden Beziehungen zwischen Personen abgebildet. Über die Intensität der Beziehungen wird nichts gesagt. In Abb. 2.9 wurde den Beziehungen jeweils ein Wert zwischen 0 und 100 zugeordnet.

Werden solche Angaben ergänzt, werden die Informationen im Graph wertvoller. Werte für Wichtungen können beispielsweise Entfernungen, Kosten, Kapazitäten oder auch domänen spezifische Priorisierungen sein.

2.3.6 Gerichtete und ungerichtete Graphen

In einem ungerichteten Graphen geht eine Beziehung immer in beide Richtungen (beispielsweise EHEPARTNER_VON). In einem gerichteten Graphen haben Beziehungen eine Richtung. Betrachtet man einen Knoten, gibt es eingehende Kanten, die auf den Knoten zeigen und ausgehende Kanten, die von dem Knoten ausgehen. Mit der Angabe von Richtung wird eine zusätzliche Informationsdimension hinzugefügt. Angenommen im linken Beispiel der Abb. 2.10 würden die Knoten Personen und die ungerichtete Kante ihre gegenseitige Freundschaft darstellen. Daraus ergibt sich, dass Person A mit Person B befreundet ist.

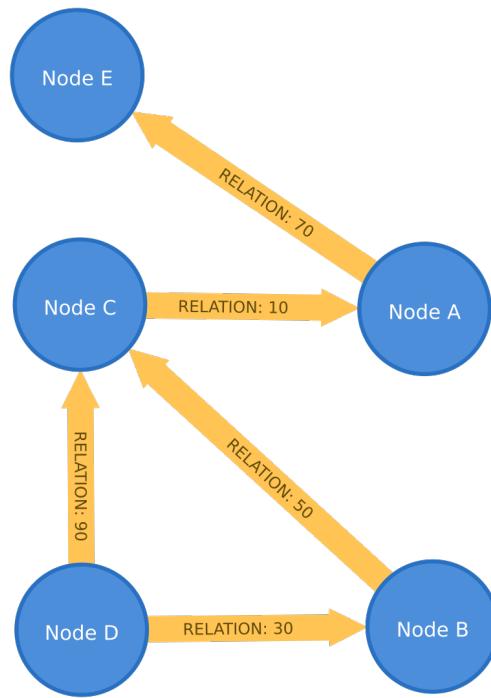


Abbildung 2.9: Gewichteter Graph (Quelle: Kuczera)

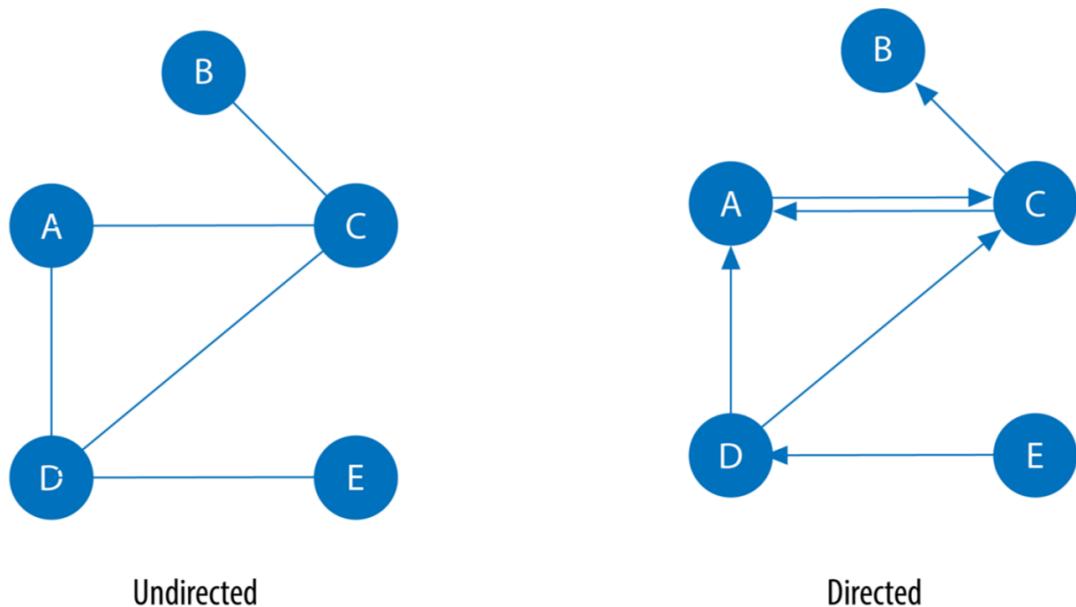


Abbildung 2.10: Gerichtete und ungerichtete Graphen (Needham/Hodler 2019, S. 21, Bild 2-7)

Erweitern wir das Beispiel für die rechte Abbildung, so dass die Freundschafts-Kanten gerichtet sind, so wird es möglich, zu zeigen, dass Person D für Person A freundschaftliche Gefühle hegt, die A aber nicht erwidert. Nehmen wir noch eine Wichtung der Freundschaftskanten hinzu, könnte man modellieren, dass die freundschaftlichen Gefühle von Person C gegenüber A mit 0,8 gewichtet sind, umgekehrt die Zuneigung von Person A zu Person C aber nur mit 0,3.

Auch bei der Modellierung von Verwandtschaft kommen gerichtete und ungerichtete Verbindungen vor.

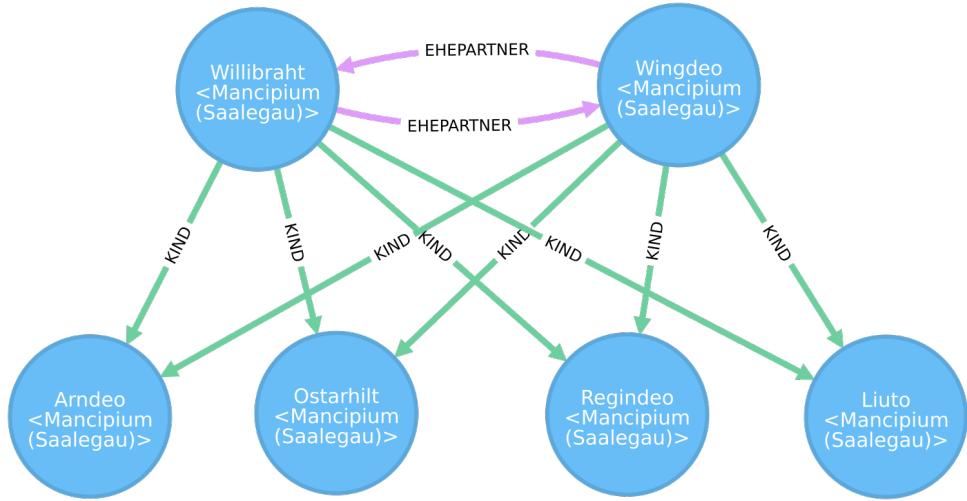


Abbildung 2.11: Verwandtschaft im Graphen mit gerichteten und ungerichteten Kanten
(Quelle: Kuczera)

In der Graphdatenbank neo4j müssen Kanten eine Richtung haben. Ungerichtete Beziehungen kann man mit zwei Kanten vom gleichen Typ aber unterschiedlicher Richtung zwischen einem Knotenpaar modellieren. Im Verwandtschaftsbeispiel Abb. 2.11 sind die zwei Personenknoten der Eltern mit zwei EHEPARTNER-Kanten unterschiedlicher Richtung verbunden. Es handelt sich um eine ungerichtete Beziehung, da es rechtlich nicht möglich ist, dass eine Person mit einer anderen Person verheiratet ist, umgekehrt aber nicht. Für die Eltern-Kind-Beziehung reicht eine Kante, da diese Beziehung hierarchisch ist.

2.4 Installation und Start

Informationen zur Installation von neo4j finden Sie auf den Dokumentationsseiten unter <https://neo4j.com/docs/operations-manual/current/installation/>. Für den normalen Nutzer empfiehlt sich die Installation von neo4j-Desktop. Unter <https://neo4j.com/blog/this-week-in-neo4j-getting-started-with-neo4j-desktop-and-browser-graphileon-personal-edition-intuitive-detections-research-with-neo4j/?ref=twitter#features-1> finden sich

Videos, in denen die Installation von neo4j-Desktop und erste Schritte im neo4j-Browser erklärt werden.

2.5 Zusammenfassung

In diesem Kapitel wurde kurz auf die Ursprünge von Graphen eingegangen und verschiedene Typen von Graphen vorgestellt. Festzuhalten ist, dass es sich in den Digitalen Geisteswissenschaften in der Regel um Mischformen der vorgestellten Graphtypen handelt. Je nach Domäne werden Graphen hoher Dichte, mit vielen Knotentypen oder auch hierarchischen Strukturen benötigt. Modelle in den Digitalen Geisteswissenschaften haben meist mehrere Knotentypen und viele Kantentypen. Dies macht wiederum die direkte Anwendung von Graph-Algorithmen schwierig, so dass Graph-Transformationen notwendig werden können.

3 Das Projekt Regesta Imperii oder “Wie suchen Onlinenutzer Regesten?”

3.1 Das Projekt Regesta Imperii

Das Projekt Regesta Imperii wurde von Johann-Friedrich Böhmer im Jahr 1829 begonnen. Ursprünglich als Vorarbeit zu den [Monumenta Germaniae Historica](#) angelegt, wurde es mit einem erweiterten Regestenkonzept bald zu einem unverzichtbaren Grundlagenwerk. In den Regesta Imperii werden Inhaltsangaben von Urkunden erstellt, die rechtlich relevante Personen, Inhalte, Orte und Sachverhalte in deutscher Sprache zusammenfassen. Zeitlich umfassen sie den Rahmen von den [Karolingern](#) (7. Jahrhundert) bis Kaiser [Maximilian](#) (gestorben 1519).



Abbildung 3.1: Urkunde zum Regest RI I n. 336, in: Regesta Imperii Online. URL: <http://www.mgh.de/bibliothek/virtueller-leseesaal/ddkar/01/?p=XXXI>

Ursprünglich von der DFG gefördert, sind die Regesta Imperii heute Teil des Bund-

Ländergeförderten Akademienprogramms und werden von der Akademie der Wissenschaften und der Literatur, Mainz, der Berlin-Brandenburgischen Akademie der Wissenschaften und der Akademie der Wissenschaften, Wien betreut.

Die Regesta Imperii arbeiten vor allem herrscherzentriert, d.h. in den Regesten muss der Herrscher eine zentrale Rolle spielen. Bei Urkundenregesten hat er selbst die Urkunde ausgestellt, bei historiographischen Regesten werden den Herrscher betreffende historische Hintergründe zusammengefasst.

Heinrich IV. - RI III,2,3 n. 1487

1103 Juni 29, Lüttich

Heinrich feiert das Fest der Apostel, wobei sich Graf Robert von Flandern im Beisein mehrerer Fürsten unterwirft, namentlich der Erzbischöfe Friedrich von Köln und Bruno von Trier, der Bischöfe Otbert von Lüttich, Burchard von Münster, Burchard von Utrecht, Herzog Heinrich von Niederlothringen sowie mehrerer Grafen.

Überlieferung/Literatur

Tagesdatum bei Ann. Patherbr. 1103 ([Scheffer-Boichorst](#) 107 f.): *in festo apostolorum Petri et Pauli; Gesta Galcheri Episcopi Cameracensis* ([SS 14](#), 202); Sigeb. Gembl. 1103 ([SS 6](#), 368); Ann. Elnon. maior. 1103 ([SS 5](#), 14); Ann. Leod., Cont. 1103 ([SS 4](#), 29); Ann. Aquens. 1103 ([SS 16](#), 685); Ann. necrol. Prum. 1103 ([SS 13](#), 223).

Kommentar

Zur Lehensterminologie in den Gesta Galcheri (*Facto palam hominio, iurat Robertus Henrico, promittit, miles domino, quia fidelis amodo*) vgl. [G a n s h o f f](#), Was ist das Lehnswesen? (³1961) 72 f. – Zum Ereigniskontext vgl. Reg. [1475](#); Heinrich V. führte bereits 1107 wieder einen Feldzug gegen Robert von Flandern; vgl. [B o s h o f](#), Bischofskirchen von Passau und Regensburg (Salier 2, 1991) 148. – Vgl. [K i l i a n](#), Itinerar 127 mit der Vermutung einer weiteren, der Unterwerfung Roberts vorangehenden Heerfahrt Heinrichs nach Flandern; [M e y e r v o n K n o n a u](#), Jbb. 5, 179 f.; [T. R e u t e r](#), Unruhestiftung (Salier 3, 1991) 324-326.

Abbildung 3.2: RI III,2,3 n. 1487, in: Regesta Imperii Online, URI: <http://www.regesta-imperii.de/id/cf75356b-bd0d-4a67-8aeb-3ae27d1dcefa>.

In der Kopfzeile des Regests in Abb. 3.2 werden der Herrscher sowie Abteilung, Band und Regestennummer genannt. Die darunterliegende Datierungszeile nennt das Ausstellungsdatum der Urkunden und den Handlungs- bzw. Ausstellungsort. Es folgt der Regestentext mit der Zusammenfassung der Urkunde, Hinweise zur Originaldatierung, die Kanzleivermerke und schließlich Angaben zur Überlieferungssituation (Gibt es eine Originalurkunde, wo liegt sie, gibt es ggf. Abschriften etc.).

3.2 Die Digitalisierung der Regesta Imperii

Im Rahmen eines von der DFG geförderten Projekts wurden die Regesta Imperii gemeinsame von der Akademie der Wissenschaften, Mainz und der Bayrischen Staatsbibliothek München von 2001 bis 2006 komplett digitalisiert. Alle seit 2006 erschienenen Regesten wurden sofort im Volltext online gestellt. Glücklicherweise hatte die Mainzer Akademie die Rechte selbst inne, so dass der Veröffentlichung als Volltext im Internet keine rechtlichen Hürden im Wege standen. Rückblickend lässt sich feststellen, dass der Absatz der gedruckten Bände nicht gelitten sondern teilweise sogar etwas zugelegt hat.

3.3 Wie suchen Online-Nutzer Regesten ?

Ende 2013 wurde das Suchverhalten der Nutzer der Online-Regestensuche im Rahmen eines Vortrages auf der Digital-Diplomatics-Konferenz in Paris in den Blick genommen.¹ Ein interessantes Ergebnis war die Häufigkeitsverteilung der Treffermengen pro Suchanfrage.

In Abb. 3.3 ist die Treffermenge in Zehnerschritten angegeben. Die hellgraue Gruppe oben rechts hat keine Treffer, die dunkelgraue Gruppe einen bis zehn Treffer, die gelbe Gruppe 11 bis 20 usw. Die lila Gruppe hat mehr als hundert Treffer. Überraschend war die große Gruppe mit über 100 Treffern. Hinzu kam, dass über 68% der Nutzer nur ein Suchwort in die Suchmaske eingegeben haben, wobei das beliebteste Suchwort *Heinrich Ende 2013* zu über 18.000 Treffern führte. Auf der Ergebnisseite hieß es dann: "Sie suchten nach *Heinrich*. Ihre Suche erzielte 18884 Treffer [...] Sie sehen die Treffer 1 bis 20."

Zusammenfassend könnte man feststellen, dass die Gruppe mit 1 bis 10 Treffern mit ihrem Ergebnis zufrieden war. 10 Regesten lassen sich gut ausdrucken und können anschließend gelesen, ausgewertet und in die eigene Forschungsarbeit einfließen. Die Gruppe mit keinem Treffer hatte möglicherweise die Suche zu sehr eingeschränkt oder einen Tippfehler beim Suchbegriff und wäre lieber in der Gruppe mit einem bis 10 Treffern. Selbstverständlich lassen sich auch 20 und mehr Treffer gut verarbeiten aber bei größeren Treffermengen steigt natürlich auch der Aufwand stark an, so dass davon auszugehen ist, dass die Nutzer kleinere, präzisere Ergebnisse bevorzugen.

Sehr gut lässt sich am Tortendiagramm auch ablesen, dass über die Hälfte unserer Nutzer vor der Suche eine genaue Vorstellung vom Ergebnis haben. Sie sind CIN-Nutzer (concrete information need). Die Gruppe mit über 100 Treffern können der Gruppe der POIN-Nutzer (problem-oriented information need) zugeordnet werden, die problemorientierte Anfragen haben. Für diese Nutzergruppe ist die aktuelle Trefferanzeige der Regestensuche unzureichend, da sie für ihre großen Treffermengen weitere Einschränkungsmöglichkeiten brauchen.²

¹Vgl. Kuczera, Andreas; Schrade, Torsten: From Charter Data to Charter Presentation: Thinking about Web Usability in the Regesta Imperii Online. Vortrag auf der Tagung ›Digital Diplomatics 2013 – What ist Diplomatics in the Digital Environment?‹ Folien: <https://prezi.com/vvacmdndthqg/from-charter-data-to-charter-presentation/>.

²Näheres dazu in Kuczera, Andreas: Digitale Perspektiven mediävistischer Quellenrecherche, in:

Distribution of Result Set Sizes

Result set sized returned for search queries between November 2012 and October 2013 (101220 queries).

- 0 Hits
- 0 - 10 Hits
- 10 - 20 Hits
- 20 - 30 Hits
- 40 - 50 Hits
- 50 - 60 Hits
- 60 - 70 Hits
- 70 - 80 Hits
- 80 - 90 Hits
- 90 - 100 Hits
- > 100 Hits

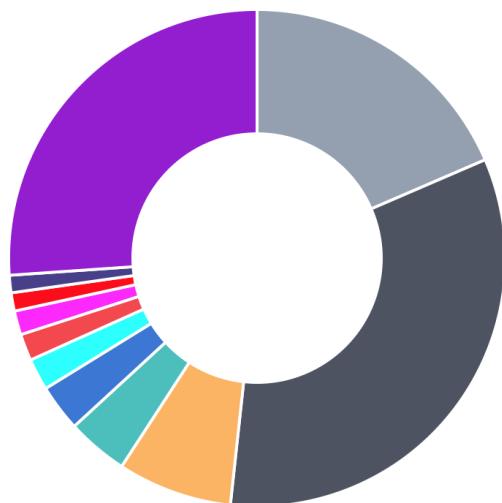


Abbildung 3.3: Treffermengen pro Suchanfragen im Jahr 2013.

3.4 Historische Netzwerkanalyse in den Registern

Im Bereich der historischen Netzwerkanalyse gab es in den letzten Jahren sehr interessante Arbeiten.³ Von Seiten der Regesta Imperii bieten sich hier vor allem die Register der Regesta Imperii als sehr interessante Quelle an. Geht man davon aus, dass alle Personen, die gemeinsam in einem Regest genannt sind, etwas miteinander zu tun haben, könnte man auf Grundlage der Registerdaten ein Personennetzwerk erstellen. Über die Qualität der Beziehungen lässt sich nichts sagen und dies schränkt die Aussage der Daten ein. Andererseits stehen sehr viele Verknüpfungen zur Verfügung.

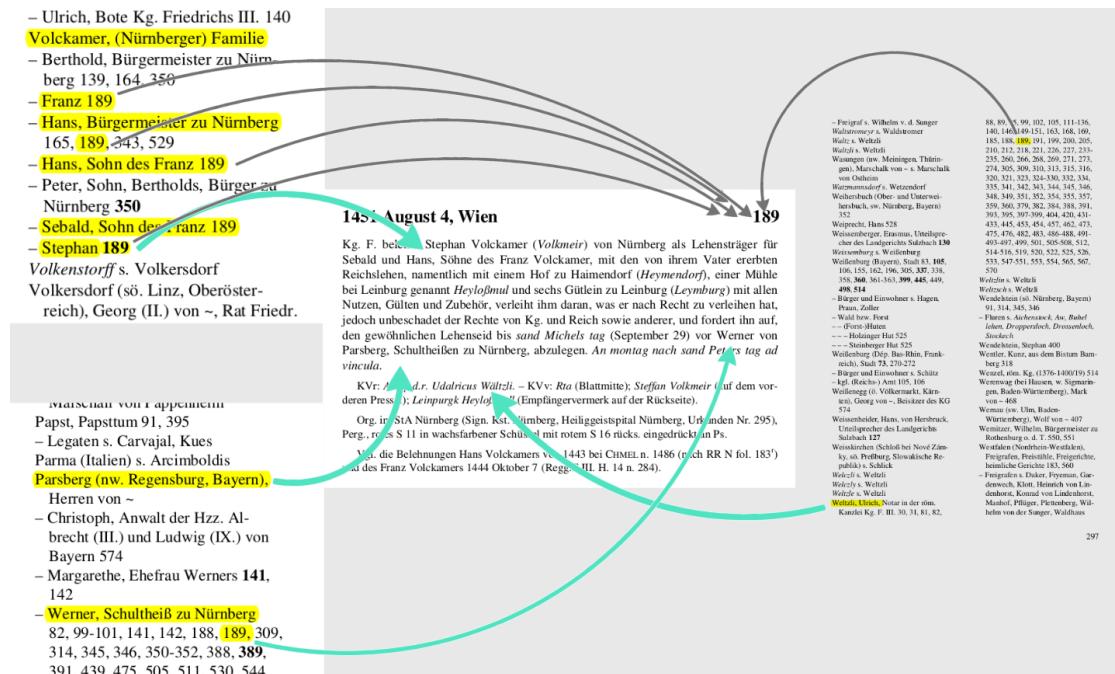


Abbildung 3.4: Registereinträge im Regest als Grundlage für ein Personennetzwerk.

Allein die Einträge in den Registern der Regesten Kaiser Friedrichs III. sind über 143.000 mal in Regesten genannt. Daraus ergeben sich dann über 460.000 1zu1-Beziehungen, vgl. Abb. 3.5.⁴

Mittelalter. Interdisziplinäre Forschung und Rezeptionsgeschichte, 18.04.2014. URL: mittelalter.hypotheses.org/3492.

³Vgl. beispielsweise Gramsch, Robert: Das Reich als Netzwerk der Fürsten - Politische Strukturen unter dem Doppelkönigtum Friedrichs II. und Heinrichs (VII.) 1225-1235. Ostfildern, 2013. Einen guten Überblick bietet das Handbuch Historische Netzwerkforschung - Grundlagen und Anwendungen. Herausgegeben von Marten Düring, Ulrich Eumann, Martin Stark und Linda von Keyserlingk. Berlin 2016.

⁴Der Cypher-Befehl zur Erstellung der 1zu1-Beziehungen lautet: `MATCH (n1:Registereintrag)-[:APPEARS_IN]->(r:Regest)<-[:APPEARS_IN]-(n2:Registereintrag) MERGE (n1)-[:KNOWS]->(n2);` Dabei werden die gerichteten KNOWS-Kanten jeweils in beide Richtungen erstellt. Mit folgendem Befehl lassen sich die KNOWS-Kanten zählen: `MATCH p=(()-[r:KNOWS]->()) RETURN count(p);` Für

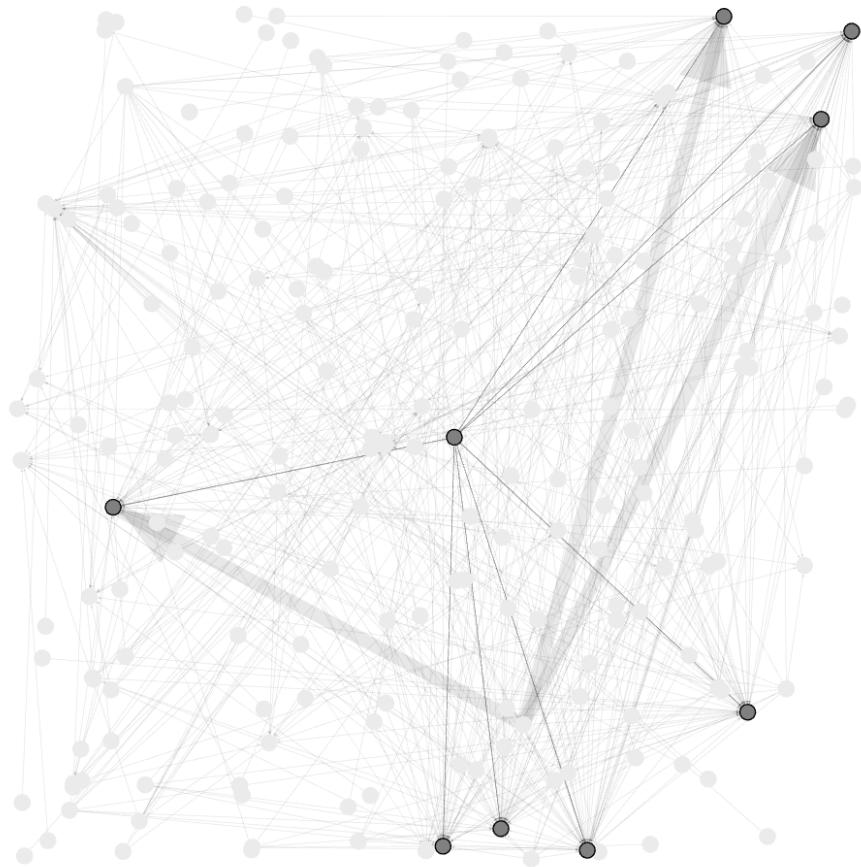


Abbildung 3.5: Ausschnitt der 1zu1-Beziehungen in Gephi.

In Abb. 3.6 sind die in den Registern des Regestenbandes von Joseph Chmel gewonnenen 1zu1-Beziehungen mit Gephi visualisiert.⁵

Bei der Analyse ergaben sich aber verschiedene Probleme. Zum einen werden in den Registern auch Kanzleibeamte genannt, die mit der eigentlichen Urkundenhandlung gar nichts zu tun hatten sondern später lediglich ihr Kürzel auf der Urkunde hinterließen. Dies mag archivgeschichtlich interessant sein, für die Urkundenhandlung ist es aber irrelevant. Ein zweites Problem ist der Aufbau des Registers, in dem Orte und Personen in einem Register zusammengefasst werden. Zum einen handelt es sich hierdurch nicht mehr um ein reines Personennetzwerk sondern um ein gemischtes Personen- und Ortsnetzwerk. Zum

die Bestimmung der 1zu1-Beziehungen muss der Wert noch durch 2 geteilt werden.

⁵Regesta chronologico-diplomatica Friderici III. Romanorum imperatoris (regis IV.) : Auszug aus den im K.K. Geheimen Haus-, Hof- und Staats-Archive zu Wien sich befindenden Registraturbüchern vom Jahre 1440 - 1493 ; nebst Auszügen aus Original-Urkunden, Manuscripten und Büchern / von Joseph Chmel, Wien 1838 und 1840.

anderen überragen die über sehr lange Zeit bestehenden Orte, die in ihrer Lebensdauer begrenzten natürlichen Personen in den Netzwerkstrukturen. Schließlich zeigte sich, dass die Algorithmen zur Netzwerkanalyse mit zeitbehafteten Daten (wie Regesten mit ihrem Ausstellungsdatum) nur schlecht umgehen konnten.

Aus Historikersicht war der Ansatz also weniger zielführend. Jedoch ergaben sich aus Modellierungssicht interessante Einblicke. Um die Netzwerke näher analysieren zu können, wurden kurze Zeitschnitte der Regesten untersucht. Hierfür musste das in Java geschriebene Programm zur Erstellung der Netzwerkdaten jedesmal umgeschrieben werden. Mein Kollege Ulli Meybohm, der das Programm damals betreute, wies mich nach dem wiederholten Umschreiben des Programms darauf hin, dass ich für meine Daten besser eine Graphdatenbank verwenden solle, beispielsweise neo4j. Erste Versuche des Imports der Registerdaten in neo4j erwiesen sich aber als sehr komplex, vgl. Abb. 3.7, obwohl das Datenmodell *Person kennt Person* eigentlich relativ einfach ist.

Schließlich ergaben Nachfragen bei neo4j, dass bei Problemen mit dem Datenmodell oft einfach ein Typ von Knoten vergessen worden sein könnte. Und tatsächlich wurden in den ersten Modellen die Regestenknoten nicht berücksichtigt. Mit den Regestenknoten im Modell war der Import schließlich mit weniger rechnerischem Aufwand möglich, vgl. Abb. 3.8.

3.5 Zusammenfassung

In diesem Kapitel wurde zunächst das Akademieprojekt Regesta Imperii vorgestellt. Seit der Anfang der 2000er Jahre erfolgten Digitalisierung stehen die Regesten unter www.regesta-imperii.de unter Creative-Commons-Lizenz frei im Internet zur Nutzung zur Verfügung. Für die Auswertung gibt es eine einfache Suchmaske und eine erweiterte Suche. Für die Jahre 2012 und 2013 wurden die Suchstrategien der Nutzer in der Online-Regestensuche untersucht und es zeigte sich, dass sich zwei Nutzungsszenarien unterscheiden lassen, von denen aber nur eines von den aktuellen Suchmasken der Regesta Imperii Online optimal bedient wird. Im zweiten Teil des Kapitels wurden die Visualisierung von Registernetzwerken und die anschließende Modellierung in Graphdatenbanken dargestellt und Nutzungs- und Auswertungsszenarien diskutiert. Im folgenden Kapitel wird die Modellierung von Regesten im Graphen detailliert erklärt.

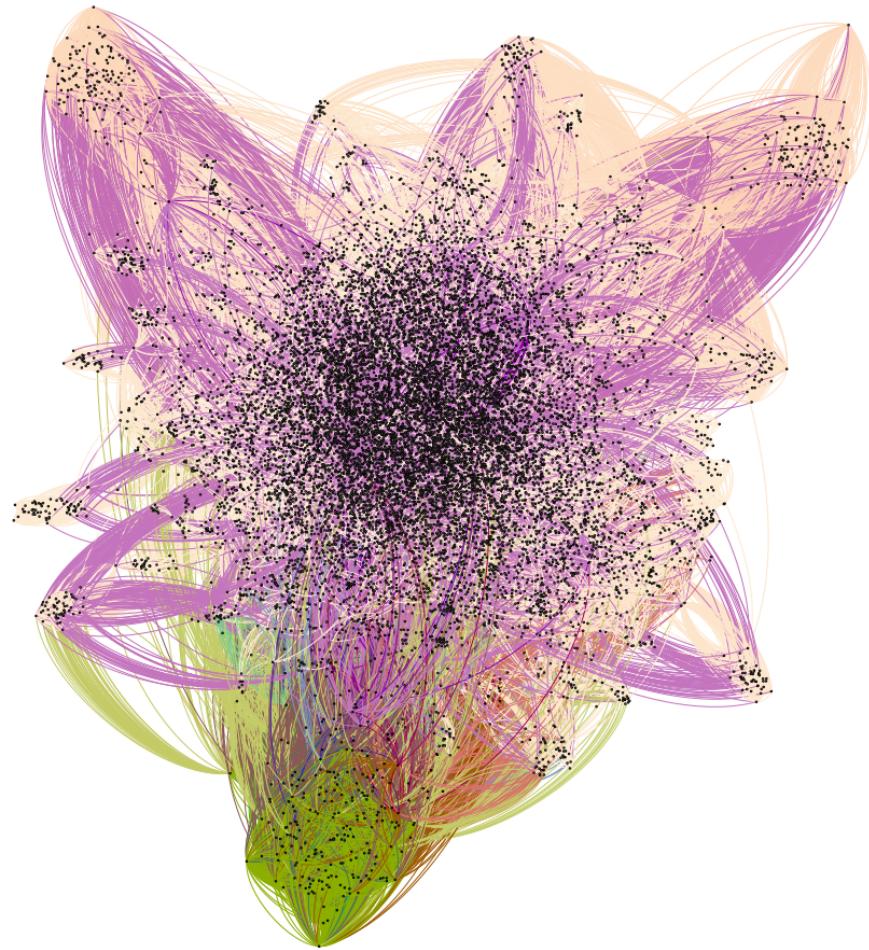


Abbildung 3.6: Personennetzwerk aus den Regestern der Regesten Chmels, erstellt mit Gephi (Quelle: Kuczera).

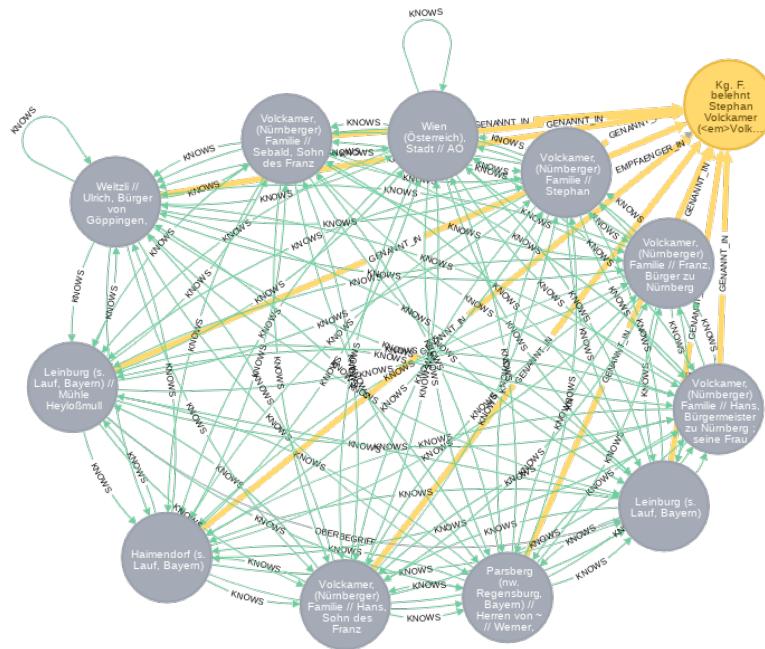


Abbildung 3.7: Regest und Registereinträge mit GENANNT_IN-Kanten und den KNOWS-Kanten.



Abbildung 3.8: Graphmodell ohne KNOWS-Kanten. Diese können bei Bedarf einfach errechnet werden.

4 Regestenmodellierung im Graphen

4.1 Wie kommen die Regesten in den Graphen

In diesem Abschnitt wird beispielhaft anhand der Regesten Kaiser Heinrichs IV. der Import der Online-Regesten in die Graphdatenbank neo4j durchgespielt.¹ Die Webseite der Regesta Imperii Online basiert auf dem Content-Management-System typo3, welches auf eine mysql-Datenbank aufbaut. In der Datenbank werden die Regesteninformationen in verschiedenen Tabellen vorgehalten. Die Webseite bietet momentan nur die Möglichkeit, die Regesten über eine REST-Schnittstelle im CEI-XML-Format herunterzuladen. Die CSV-Version, die sich für den Import in die Graphdatenbank anbietet findet sich auf Github: <https://github.com/kuczera/GraphReader/raw/master/data/RegH4.csv>. Unter <https://github.com/kuczera/GraphReader/raw/master/data/GraphReaderCypher.txt> ist eine Textdatei mit den Listings aller cypher-Befehle erhältlich.

In der in Abb. 4.1 dargestellten CSV-Datei finden sich die oben erläuterten einzelnen Elemente der Regesten in jeweils eigenen Spalten. Die Spaltenüberschrift gibt Auskunft zum Inhalt der jeweiligen Spalte.

¹Die Einrichtung der Graphdatenbank Neo4j wird erläutert unter <https://neo4j.com/docs/operations-manual/current/installation/>. Die Regesten Kaiser Heinrichs IV. umfassen folgende Bände: Böhmer, J. F., Regesta Imperii III. Salisches Haus 1024-1125. Tl. 2: 1056-1125. 3. Abt.: Die Regesten des Kaiserreichs unter Heinrich IV. 1056 (1050) - 1106. 1. Lief.: 1056 (1050) – 1065, bearb. von Struve, Tilman - Köln (u.a.) (1984). Böhmer, J. F., Regesta Imperii III. Salisches Haus 1024-1125. Tl. 2: 1056-1125. 3. Abt.: Die Regesten des Kaiserreichs unter Heinrich IV. 1056 (1050) - 1106. 2. Lief.: 1065–1075, bearb. von Struve, Tilman unter Mitwirkung von Lubich, Gerhard und Jäckel, Dirk - Köln (u.a.) (2010). Böhmer, J. F., Regesta Imperii III. Salisches Haus 1024-1125. Tl. 2: 1056-1125. 3. Abt.: Die Regesten des Kaiserreichs unter Heinrich IV. 1056 (1050) - 1106. 3. Lief.: 1076–1085, bearb. von Lubich, Gerhard nach Vorarbeiten von Struve, Tilman unter Mitwirkung von Jäckel, Dirk - Köln (u.a.) (2016). Böhmer, J. F., Regesta Imperii III. Salisches Haus 1024-1125. Tl. 2: 1056-1125. 3. Abt.: Die Regesten des Kaiserreichs unter Heinrich IV. 1056 (1050) - 1106. 4. Lief.: 1086–1105/06, bearb. von Lubich, Gerhard nach Vorarbeiten von Brauch, Daniel unter Mitwirkung von Weber, Matthias - Köln (u.a.) (2016). Böhmer, J. F., Regesta Imperii III. Salisches Haus 1024-1125. Tl. 2: 1056-1125. 3. Abt.: Die Regesten des Kaiserreichs unter Heinrich IV. 1056 (1050) - 1106. 5. Lief.: Die Regesten Rudolfs von Rheinfelden, Hermanns von Salm und Konrads (III.). Verzeichnisse, Register, Addenda und Corrigenda, bearbeitet von Lubich, Gerhard unter Mitwirkung von Junker, Cathrin; Klocke, Lisa und Keller, Markus - Köln (u.a.) (2018).

uid	start_date	end_date	identifier	regst_persistent_id	title	date_string	locality_string	summary	archival_history
20053	1050-11-11	1050-11-11	RI III,2,3 n. 1	1 1050	Heinrich IV.	1050	(Georg?)	Während eines Wettreitens	1050
20054	1050-12-25	1050-12-25	RI III,2,3 n. 2	2 1050	Heinrich IV.	1050 Dezern. Wettreit	Heinrichs Vg Herim. Aug. 1051		
20055	1051-02-02	1051-02-02	RI III,2,3 n. 2	3 1051	Heinrich IV.	1051 Februar	Augsburg	Heinrich folg. Herim. Aug. 1051	
20056	1051-03-04	1051-03-04	RI III,2,3 n. 4	4 1051	Heinrich IV.	1051 März (v Speyer)		Heinrich folg. Herim. Aug. 1051	
20057	1051-03-31	1051-03-31	RI III,2,3 n. 5	5 1051	Heinrich IV.	1051 März 3 Köln		Heinrich emi Herim. Aug. 1051	
20058	1051-11-12	1051-11-12	RI III,2,3 n. 6	6 1051	Heinrich IV.	1051 Novem Regensburg		Während des Umgangsfürzuges	
20059	1051-12-25	1051-12-25	RI III,2,3 n. 7	7 1051	Heinrich IV.	1051 Dezem Goslar		Weihnachtsf Herim. Aug. 1052	
20060	1052-03-05	1052-03-05	RI III,2,3 n. 8	8 1052	Heinrich IV.	1052 März 5 Kaiserswert		Heinrich inte <link href="http://opac.re	
20061	1052-03-27	1052-03-27	RI III,2,3 n. 9	9 1052	Heinrich IV.	1052 März 2 Goslar		Heinrichs wil <link href="http://opac.re	
20062	1052-06-07	1052-06-07	RI III,2,3 n. 10	10 1052	Heinrich IV.	1052 Juni (7 Zürich		Heinrich folg. Herim. Aug. 1052	
20063	1052-09-01	1052-09-01	RI III,2,3 n. 11	11 1052	Heinrich IV.	1052 Septe –		Heinrichs Elf Herim. Aug. 1052	
20064	1052-12-25	1052-12-25	RI III,2,3 n. 12	12 1052	Heinrich IV.	1052 Dezem Worms		Weihnachtsf Herim. Aug. 1053	
14	1053-01-03	1053-01-03	RI III,2,3 n. 13	13 1053	Heinrich IV.	1053 Januar vor de Tribur		Heinrich wir. Herim. Aug. 1053	
20066	1053-01-25	1053-01-25	RI III,2,3 n. 14	14 1053	Heinrich IV.	1053 Januar Überfahrt		Heinrichs Wil. Aug. 1053	
20067	1054-04-03	1054-04-03	RI III,2,3 n. 15	15 1054	Heinrich IV.	1054 April (u Mainz)		Heinrich folg. Herim. Aug. 1054	
20068	1054-05-29	1054-05-29	RI III,2,3 n. 16	16 1054	Heinrich IV.	1054 Mai 29 (Goslar?)		Heinrich inte <link href="http://opac.re	
20069	1054-05-31	1054-05-31	RI III,2,3 n. 17	17 1054	Heinrich IV.	1054 Mai 31 Goslar		Heinrich inte <link href="http://opac.re	
20070	1054-07-17	1054-07-17	RI III,2,3 n. 18	18 1054	Heinrich IV.	1054 Juli 17 Aachen		Heinrich wir Lampert 1054 <link	
20071	1054-07-17	1054-07-17	RI III,2,3 n. 19	19 1054	Heinrich IV.	1054 (nach c Aachen?)		Vermutlich ähnlich Heinrichs I	
20072	1054-11-17	1054-11-17	RI III,2,3 n. 20	20 1054	Heinrich IV.	1054 Novem Mainz		Heinrichs wil <link href="http://opac.re	
20073	1054-11-17	1054-11-17	RI III,2,3 n. 21	21 1054	Heinrich IV.	1054 Novem Mainz		Heinrichs Wil <link href="http://opac.re	
20074	1054-12-25	1054-12-25	RI III,2,3 n. 22	22 1054	Heinrich IV.	1054 Dezem Goslar		Weihnachtsf Ann Altah. 1055 (
20075	1055-01-16	1055-01-16	RI III,2,3 n. 23	23 1055	Heinrich IV.	1055 Januar Quedlinburg		Heinrichs Wil <link href="http://opac.re	
20076	1055-03-03	1055-03-03	RI III,2,3 n. 24	24 1055	Heinrich IV.	1055 März 3 Regensburg		Heinrichs Wil <link href="http://opac.re	
20077	1055-03-06	1055-03-06	RI III,2,3 n. 25	25 1055	Heinrich IV.	1055 März 6 Regensburg		Heinrichs Wil <link href="http://opac.re	
20078	1055-03-12	1055-03-12	RI III,2,3 n. 26	26 1055	Heinrich IV.	1055 März 1 Utting(?)		Heinrichs wil <link href="http://opac.re	

Abbildung 4.1: Regesten als CSV-Datei

4.1.1 Import mit dem LOAD CSV-Befehl

Mit dem Befehl `LOAD CSV` können die CSV-Dateien mit den Regesten in die Graph-datenbank neo4j importiert werden.² Hierfür muss die Datenbank aber Zugriff auf die CSV-Daten haben. Dies ist einerseits über den im Datenbankverzeichnis vorhandene Ordner `import` oder über eine URL, unter der die CSV-Datei abrufbar ist, möglich. Da sich die einzelnen Zugriffswege auf den `import`-Ordner von Betriebssystem zu Betriebssystem unterscheiden, wird hier beispielhaft der Import über eine URL vorgestellt. Hierfür wird ein Webserver benötigt, auf den man die CSV-Datei hochlädt und sich anschließend die Webadresse für den Download der Datei notiert. Für die hier vorgestellten Beispiele werden die Daten über github bereitgestellt: <https://github.com/kuczera/GraphReader/raw/master/data/RegH4.csv>.

4.1.2 Regestenmodellierung im Graphen

Mit dem `LOAD CSV`-Befehl stehen die Informationen der Regestentabelle nun für die weitere Verarbeitung zur Verfügung. Nun muss festgelegt werden, wie diese Informationen im Graphen modelliert werden sollen. Daher wird im nächsten Schritt das Modell der Regesten im Graphen vorgestellt (Abb. 4.2).

In den Abbildungen 4.2 und 4.3 finden sich beispielhaft das Regest RI III,2,3 Nr. 1487, einmal in der Ansicht der Onlineregesten und in der zweiten Abbildung als Modell im Graphen (neben anderen Regesten).

Die gelben Knoten sind die Regesten. Aus den Angaben des Regests werden mit dem o.a. Befehl noch ein Datumsknoten und ein Ortsknoten erstellt. Mit dem ersten `CREATE`-Befehl

²Zu Installation und ersten Schritten von neo4j vgl. in der Einleitung den Abschnitt zu Installation und Start.

Heinrich IV. - RI III,2,3 n. 1487

1103 Juni 29, Lüttich

Heinrich feiert das Fest der Apostel, wobei sich Graf Robert von Flandern im Beisein mehrerer Fürsten unterwirft, namentlich der Erzbischöfe Friedrich von Köln und Bruno von Trier, der Bischöfe Otbert von Lüttich, Burchard von Münster, Burchard von Utrecht, Herzog Heinrich von Niederlothringen sowie mehrerer Grafen.

Überlieferung/Literatur

Tagesdatum bei Ann. Patherbr. 1103 ([Scheffer-Bochorst](#) 107 f.); *in festo apostolorum Petri et Pauli; Gesta Galcheri Episcopi Cameracensis* (SS 14, 202); Sigeb. Gembl. 1103 (SS 6, 368); Ann. Elnon. maior. 1103 (SS 5, 14); Ann. Leod. Cont. 1103 (SS 4, 29); Ann. Aquens. 1103 (SS 16, 685); Ann. necrol. Prum. 1103 (SS 13, 223).

Kommentar

Zur Lehensterminologie in den Gesta Galcheri (*Facto palam hominio, iurat Robertus Henrico, promittit, miles domino, quia fidelis amodo*) vgl. [G a n s h o f f](#), Was ist das Lehnswesen? (1961) 72 f. – Zum Ereigniskontext vgl. Reg. 1475; Heinrich V. führte bereits 1107 wieder einen Feldzug gegen Robert von Flandern; vgl. [B o s h o f](#), Bischofskirchen von Passau und Regensburg (Salier 2, 1991) 148. – Vgl. [Kili an](#), Itinerar 127 mit der Vermutung einer weiteren, der Unterwerfung Roberts vorangehenden Heerfahrt Heinrichs nach Flandern; [Meyer von Knonau](#), Jbb. 5, 179 f.; [T. Reuter](#), Unruhestiftung (Salier 3, 1991) 324–326.

Abbildung 4.2: RI III,2,3 n. 1487, in: Regesta Imperii Online, URI: <http://www.regesta-imperii.de/id/cf75356b-bd0d-4a67-8aeb-3ae27d1dcefa>.

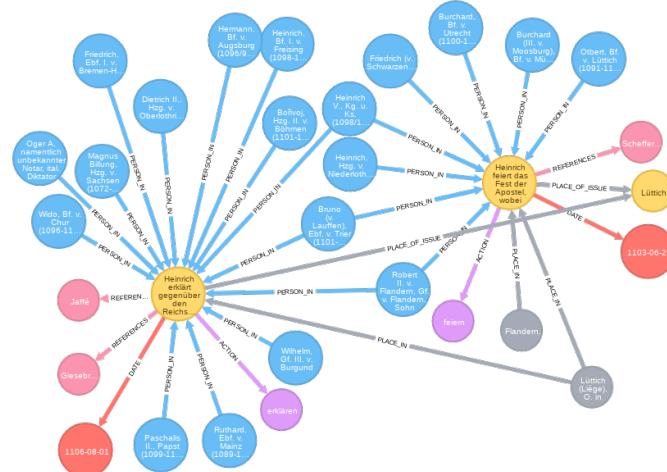


Abbildung 4.3: Das Regest im Graphen.

werden die Regesten erstellt. Die MERGE-Befehle erzeugen ergänzende Knoten für die Datumsangaben und die Ausstellungsorte. Nun ist es aber so, dass Ausstellungsort und Ausstellungsdatum mehrfach vorkommen können. Daher wird hier nicht der CREATE-Befehl sondern der MERGE-Befehl verwendet. Dieser funktioniert wie der CREATE-Befehl, prüft aber vorher, ob in der Datenbank ein solcher Knoten schon existiert. Falls es ihn noch nicht gibt, wird er erzeugt, wenn es ihn schon gibt, wird er der entsprechenden Variable zugeordnet. Anschließend werden die Kanten zwischen Regestenknoten und Ausstellungsortsknoten sowie Regestenknoten und Datumsknoten erstellt. In der folgenden Tabelle werden die einzelnen Befehle dargestellt und kommentiert.

4.1.3 Indexe Erstellen

Bevor nun mit dem Import begonnen wird, ist es für die Beschleunigung des Importprozesses von Vorteil vorher Indexe für häufig genutzte Properties zu erstellen.

```
// vorab Index erzeugen -> Import wird schneller
CREATE INDEX ON :Regesta(ident);
CREATE INDEX ON :Regesta(regnum);
CREATE INDEX ON :Regesta(persistentIdentifier);
CREATE INDEX ON :Regesta(registerId);
CREATE INDEX ON :Regesta(heftId);
CREATE INDEX ON :Regesta(placeOfIssue);
CREATE INDEX ON :Regesta(origPlaceOfIssue);
CREATE INDEX ON :Date(startDate);
CREATE INDEX ON :Place(original);
CREATE INDEX ON :Place(normalizedGerman);
CREATE INDEX ON :Lemma(lemma);
CREATE INDEX ON :Literature(literatur);
CREATE INDEX ON :Reference(reference);
CREATE INDEX ON :IndexEntry(registerId);
CREATE INDEX ON :IndexEntry(nodeId);
CREATE INDEX ON :Regesta(latLong);
CREATE INDEX ON :IndexPlace(registerId);
CREATE INDEX ON :IndexEvent(registerId);
CREATE INDEX ON :IndexPerson(registerId);
```

4.1.4 Erstellen der Regestenknoten

Mit dem folgenden Cypher-Query werden die Regestenknoten in der Graphdatenbank erstellt:

```
// Regestenknoten erstellen
LOAD CSV WITH HEADERS FROM "https://github.com/kuczera/GraphReader/raw/master/data/RegH4
CREATE (r:Regesta {regid:line.persistentIdentifier, text:line.summary,
```

```

archivalHistory:line.archival_history, date:line.date_string,
ident:line.identifier, regnum:line.regnum,
origPlaceOfIssue:line.locality_string, startDate:line.start_date,
endDate:line.end_date})
MERGE (d:Date {startDate:line.start_date, enddate:line.end_date})
MERGE (r)-[:DATE]->(d)
RETURN count(r);

```

Im Folgenden werden die einzelnen Teile des Import-Befehls erläutert:

Befehl	Variablen	Bemerkungen
LOAD CSV WITH HEADERS FROM "htt- ps://github.com/kuczera/ GraphReader/raw/master/ data/RegH4.csv" AS line	line	Import der CSV-Dateien. Es wird jeweils eine Zeile an die Variable line weitergegeben
CREATE(r:Regesta {re- gid:line.persistentIdentifier, text:line.summary, archivalHisto- ry:line.archival_history, date:line.date_string etc.}	line.persistent_ identifier, line.summary etc.	Erstellung des Regestenknotens. Für die weiteren Befehle steht der neu erstellt Regestenknoten unter der Variable r zur Verfügung.
MERGE (d:Date {startDate:line.start_date, enddate:line.end_date})	line.start_date und line.end_date	Es wird geprüft, ob ein Datumsknoten mit der Datumsangabe schon existiert, falls nicht, wird er erstellt. In jedem Fall steht anschließend der Datumsknoten unter der Variable d zur Verfügung.
MERGE (r)-[:HAT_DATUM]->(d)	(r) ist der Regestenknoten, (d) ist der Datumsknoten	Zwischen Regestenknoten und Datumsknoten wird eine HAT_DATUM-Kante erstellt.

4.1.5 Erstellen der Ausstellungsorte

In den Kopfzeilen der Regesten ist, soweit bekannt, der Ausstellungsort der Urkunde vermerkt. Im Rahmen der Arbeiten an den Regesta Imperii Online wurden diese Angaben zusammengestellt und soweit möglich die Orte identifiziert, so dass diese Angabe nun beim Import der Regesten in den Graphen berücksichtigt werden kann. Insgesamt befinden sich in den Regesta Imperii über 12.000 verschiedene Angaben für Ausstellungsorte, wobei sie sich aber auch teilweise auf den gleichen Ort beziehen können (Wie z.B. Aachen, Aquisgrani, Aquisgradi, Aquisgranum, coram Aquisgrano etc.). Allein mit der

Identifizierung der 1.000 häufigsten Ortsangaben konnte schon die überwiegende Mehrzahl der Ausstellungsorte georeferenziert werden. Die Daten zur Ortsidentifizierung liegen auch als CSV-Datei vor.

Mit dem folgenden Cypher-Query werden die Ausstellungsorte in die Graphdatenbank importiert:

```
// RI-Ausstellungsorte-geo erstellen
LOAD CSV WITH HEADERS FROM "https://github.com/kuczera/GraphReader/raw/master/data/RI_Ori
AS line
WITH line
WHERE line.Lat IS NOT NULL
AND line.normalisiertDeutsch IS NOT NULL
MATCH (r:Regesta {origPlaceOfIssue:line.Original})
MERGE (p:Place {normalizedGerman:line.normalisiertDeutsch,
    longitude:line.Long, latitude:line.Lat})
WITH r, p, line
MERGE (r)-[rel:PLACE_OF_ISSUE]->(p)
SET p.wikidataId = line.wikidataId
SET p.name = line.name
SET p.gettyId = line.GettyId
SET p.geonamesId = line.GeonamesId
SET rel.original = line.Original
SET rel.alternativeName = line.Alternativname
SET rel.commentary = line.Kommentar
SET rel.allocation = line.Zuordnung
SET rel.state = line.Lage
SET rel.certainty = line.Sicherheit
SET rel.institutionInCity = line.InstInDerStadt
RETURN count(p);
```

Da der Import-Query etwas komplexer ist, wird er im folgenden näher erläutert. Nach dem LOAD CSV WITH HEADERS FROM-Befehl wird zunächst überprüft, ob der jeweils eingelesene Eintrag in der Spalte line.lat und in der Spalte line.normalisiertDeutsch Einträge hat. Ist dies der Fall, wird überprüft, ob es einen Regestenknoten gibt, der einen Ausstellungsorteintrag hat, der der Angabe in der Spalte Original entspricht. Diese Auswahl ist notwendig, da in der Tabelle die Ausstellungsorte der gesamten Regesta Imperii enthalten sind. Für diesen Import sollen aber nur jene angelegt werden, die für die Regesten Kaiser Heinrichs IV. relevant sind. Mit dem MERGE-Befehl wird der Place-Knoten erstellt (falls es ihn nicht schon gibt) und anschließend mit dem Regestenknoten verknüpft. Schließlich werden noch weitere Details der Ortsangabe im Place-Knoten und in den PLACE_OF_ISSUE-Kanten ergänzt.

4.1.6 Koordinaten der Ausstellungsorte

Mit dem folgenden Query werden die Koordinatenangaben zu Höhen- und Breitengraden der Ausstellungsorte (Place-Knoten), die in den Propertys `latitude` und `longitude` abgespeichert sind, in der neuen Property `LatLong` zusammengefasst und in `point`-Werte umgewandelt. Seit Version 3 kann neo4j mit diesen Werten Abstandsberechnungen durchführen (Mehr dazu siehe unten bei den Auswertungen).

```
// Regesten und Ausstellungsorte mit Koordinaten der Ausstellungsorte versehen
MATCH (r:Regesta)-[:PLACE_OF_ISSUE]->(o:Place)
SET r.latLong = point({latitude: tofloat(o.latitude),
    longitude: tofloat(o.longitude)})
SET o.latLong = point({latitude: tofloat(o.latitude),
    longitude: tofloat(o.longitude)})
SET r.placeOfIssue = o.normalizedGerman
SET r.latitude = o.latitude
SET r.longitude = o.longitude;
```

4.1.7 Ausstellungsdatum

In den Regesta Imperii Online sind die Datumsangaben der Regesten iso-konform im Format `JJJJ-MM-TT` (also Jahr-Monat-Tag) abgespeichert. Neo4j behandelt diese Angaben aber als String. Um Datumsberechnungen durchführen zu können, müssen die Strings in neo4j-interne Datumswerte umgerechnet werden. Der Cypher-Query hierzu sieht wie folgt aus:

```
// Date in neo4j-Datumsformat umwandeln
MATCH (n:Regesta)
SET n.isoStartDate = date(n.startDate);
MATCH (n:Regesta)
SET n.isoEndDate = date(n.endDate);
MATCH (d:Date)
SET d.isoStartDate = date(d.startDate);
MATCH (d:Date)
SET d.isoEndDate = date(d.endDate);
```

Zunächst werden mit dem `MATCH`-Befehl alle Regestenknoten aufgerufen. Anschließend wird für jeden Regestenknoten aus der String-Property `startDate` die Datumsproperty `isoStartDate` berechnet und im Regestenknoten abgespeichert. Mit Hilfe der Property können dann Datumsangaben und Zeiträume abgefragt werden (Beispiel hierzu unten in der Auswertung).

4.2 Exkurs 1: Herrscherhandeln in den Regesta Imperii

Bisher wurden beim Import der Regesten in den Graphen nur die in den Online-Regesten bereits angelegten Angaben importiert. Im folgenden Schritt werden nun in einem kleinen Exkurs die Regestentexte selbst analysiert und anschließend die Graphdatenbank um eine weitere Informationsebene ergänzt. Regesten sind in ihrer Struktur stark formalisiert. Meist wird mit dem ersten Verb im Regest das Herrscherhandeln beschrieben. Um dies auch digital auswerten zu können, haben wir in einem kleinen Testprojekt mit Hilfe des [Stuttgart-München Treetaggers](#)³ aus jedem Regest das erste Verb extrahiert und normalisiert. Die Ergebnisse sind einsehbar unter <https://github.com/kuczera/GraphReader/raw/master/data/ReggH4-Verben.csv>. Diese Tabelle wird mit dem folgenden Cypher-Query in die Graphdatenbank eingelesen.

```
// ReggH4-Herrscherhandeln
LOAD CSV WITH HEADERS FROM "https://github.com/kuczera/GraphReader/raw/master/data/ReggH4-Verben.csv"
AS line FIELDTERMINATOR ','
MATCH (r:Regesta{ident:line.regid})
MERGE (l:Lemma{lemma:line.Lemma})
MERGE (r)-[:ACTION]->(l);
```

Dabei wird zunächst mit dem MATCH-Befehl das jeweilige Regest gesucht, anschließend mit dem MERGE-Befehl der Lemma-Knoten für das Herrscherhandeln angelegt (falls noch nicht vorhanden) und schließlich der Regesta-Knoten mit dem Lemma-Knoten über eine ACTION-Kante verbunden. In Abb. 4.4 ist ein Ausschnitt mit Regesten und den verknüpften Lemmaknoten dargestellt.

4.3 Zitationsnetzwerke in den Regesta Imperii

In vielen Online-Regesten ist die zitierte Literatur mit dem [Regesta-Imperii-Opac](#) verlinkt. Da es sich um URLs handelt, sind diese Verweise eindeutig. Andererseits lassen sie sich mit regulären Ausdrücken aus den Regesten extrahieren. Mit folgendem Query werden aus den Überlieferungsteilen der Regesten die mit dem Opac verlinkten Literaturangaben extrahiert und jede Literaturangabe als Reference-Knoten angelegt. Für den folgenden Befehl muss die APOC-Bibliothek in neo4j installiert sein: https://neo4j-contrib.github.io/neo4j-apoc-procedures/#_installation_with_neo4j_desktop.

```
// ReggH4-Literaturnetzwerk erstellen
MATCH (reg:Regesta)
WHERE reg.archivalHistory CONTAINS "link"
UNWIND apoc.text.regexGroups(reg.archivalHistory,
"<link (\\"S+)>(\\"S+)</link>") as link
MERGE (ref:Reference {url:link[1]})
ON CREATE SET ref.title=link[2]
```

³Zum Treetagger vgl. <http://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/>.

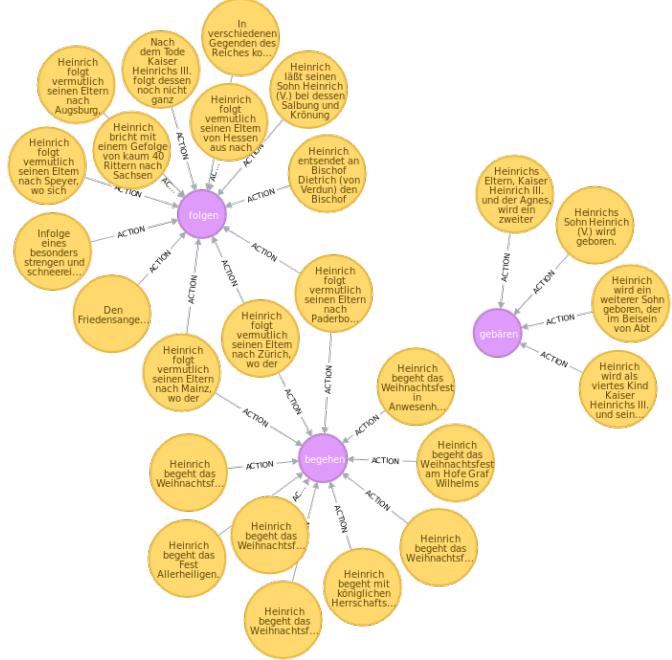


Abbildung 4.4: Herrscherhandeln im Graphen.

```
MERGE (:reg)-[:REFERENCES]->(:ref);
```

Da dies mit dem MERGE-Befehl geschieht, wird in der Graphdatenbank jeder Literaturtitel nur einmal angelegt. Anschließend werden die Reference-Knoten mit den Regesten über REFERENCES-Kanten verbunden. Zu den Auswertungsmöglichkeiten vgl. unten den Abschnitt zu den [Auswertungsperspektiven](#).

4.4 Import der Registerdaten in die Graphdatenbank

4.4.1 Vorbereitung der Registerdaten

Register spielen für die Erschließung von gedrucktem Wissen eine zentrale Rolle, da dort in alphabetischer Ordnung die im Werk vorkommenden Entitäten (z.B. Personen und Orte) hierarchisch gegliedert aufgeschlüsselt werden. Für die digitale Erschließung der Regesta Imperii sind Register von zentraler Bedeutung, da mit ihnen die in den Regesten vorkommenden Personen und Orte bereits identifiziert vorliegen. Für den Import in die Graphdatenbank wird allerdings eine digitalisierte Fassung des Registers benötigt. Im Digitalisierungsprojekt Regesta Imperii Online wurden Anfang der 2000er Jahre auch die gedruckt vorliegenden Register digitalisiert. Sie dienen nun als Grundlage für die digitale Registererschließung der Regesta Imperii. Im hier gezeigten Beispiel werden die Regesten Kaiser Heinrichs IV. und das dazugehörige Register importiert. Da der letzte Regestenband der Regesten Kaiser Heinrichs IV. mit dem Gesamtregister erst vor kurzem

gedruckt wurde, liegen hier aktuelle digitale Fassung von Registern und Regesten vor. Die für den Druck in Word erstellte Registerfassung wird hierfür zunächst in eine hierarchisch gegliederte XML-Fassung konvertiert (vgl. Abb. 4.5), damit die Registerhierarchie auch maschinenlesbar abgelegt ist.

```

<Stufe0 id="H4P00005">
    <Inhalt>Abiram, biblische Gestalt, Sohn Eliabs, Bruder Dathans, Auflehner gegen
        Moses</Inhalt>
    <Regestennummer>
        <r>762</r>
    </Regestennummer>
</Stufe0>
<Stufe0 id="H4P00006">
    <Inhalt>AC → Gottschalk v. Aachen</Inhalt>
</Stufe0>
<Stufe0 id="H4P00007">
    <Inhalt>Achalmer, Adelsgeschlecht aus Schwaben</Inhalt>
    <Regestennummer>
        <r>363</r>
    </Regestennummer>
    <Stufe1>
        <Inhalt><vw/>- Liutold, Gf. v. Achalm</Inhalt>
    </Stufe1>
    <Stufe1>
        <Inhalt><vw/>- Werner (v. Achalm), Bf. II. v. Straßburg</Inhalt>
    </Stufe1>
    <Stufe1>
        <Inhalt><vw/>- Williberga</Inhalt>
    </Stufe1>
</Stufe0>
```

Abbildung 4.5: Ausschnitt aus dem XML-Register der Regesten Heinrichs IV.

In der XML-Fassung sind die inhaltlichen Bereiche und die Abschnitte für die Regestennummern jeweils extra in die Tags `<Inhalt>` und `<Regestennummer>` eingefasst. Innerhalb des Elements `<Regestennummer>` ist dann nochmal jede einzelne Regestennummer in `<r>`-Tags eingefasst. Die aus dem gedruckten Register übernommenen Verweise sind durch ein leeres `<vw/>`-Element gekennzeichnet.

Die in XML vorliegenden Registerdaten werden anschließend mit Hilfe von TuStep in einzelne CSV-Tabellen zerlegt.

6	H4P00005	Abiram, biblische Gestalt, Sohn Eliabs, Bruder Dathans, Auflehner gegen Moses		
7	H4P00006	AC → Gottschalk v. Aachen		
8	H4P00007	Achalmer, Adelsgeschlecht aus Schwaben		
9	H4P00008	Aczo, Sohn Rudolf Rubias, Bruder Attos		
10	H4P00009	Adalbero, Mgf. d. karantanischen Mark		
11	H4P00010	Adalbero, Gf. v. Ebersberg, Gem. Richildes		
12	H4P00011	Adalbero, Edelfreier		
13	H4P00012	Adalbero A (AA), namentlich unbekannter Bamberger Diktator		

Abbildung 4.6: Ausschnitt der Entitätentabelle des Registers der Regesten Heinrichs IV.

In einer Tabelle werden alle Entitäten aufgelistet und jeweils mit einer ID versehen (Abb. 4.6).

ID	regnum	regnum2	name1	name2
H4P00001	805	805		<i>A.</i>,
H4P00002	1517	1517		<i>A.</i>,
H4P00003	1519	1519		<i>A.</i>, <i>centurio</i>
H4P00005	762	762		Abiram, biblische Gestalt, Sohn Eliabs, Bruder Dathans, Auflehner gegen Moses
H4P00007	363	363		Achalmer, Adelsgeschlecht aus Schwaben
H4P00008	1056	1056		Aczo, Sohn Rudolf Rubias, Bruder Attos
H4P00009	714	714		Adalbero, Mgf. d. karantanischen Mark
H4P00010	78	78		Adalbero, Gf. v. Ebersberg, Gem. Richildes
H4P00011	331	331		Adalbero, Edelfreier
H4P00012	666	666		Adalbero A (AA), namentlich unbekannter Bamberger Diktator
H4P00012	717	717		Adalbero A (AA), namentlich unbekannter Bamberger Diktator
H4P00012	959	959		Adalbero A (AA), namentlich unbekannter Bamberger Diktator
H4P00012	1400	1400		Adalbero A (AA), namentlich unbekannter Bamberger Diktator
H4P00012	1403	1403		Adalbero A (AA), namentlich unbekannter Bamberger Diktator
H4P00012	1420	1420		Adalbero A (AA), namentlich unbekannter Bamberger Diktator
H4P00012	1481	1481		Adalbero A (AA), namentlich unbekannter Bamberger Diktator
H4P00012	1485	1485		Adalbero A (AA), namentlich unbekannter Bamberger Diktator

Abbildung 4.7: Ausschnitt der Verknüpfungstabelle des Registers der Regesten Heinrichs IV.

In der anderen Tabelle werden die Verknüpfungen zwischen Registereinträgen und den Regesten aufgelistet (Abb. 4.7). Der Registereintrag Adalbero kommt in mehreren Regesten vor. Da das Register der Regesten Heinrichs IV. nur zwei Hierarchiestufen enthält, in denen beispielsweise verschiedene Amtsphasen ein und derselben Person unterschieden werden, wurden diese beim Import zusammengefasst.⁴ Damit gibt es pro Person jeweils nur einen Knoten.

4.4.2 Import der Registerdaten in die Graphdatenbank

Im Gegensatz zu den Regesten Kaiser Friedrichs III., bei denen Orte und Personen in einem Register zusammengefasst sind, haben die Regesten Kaiser Heinrich IV. getrennte Orts- und Personenregister. Die digitalisierten Registerdaten können als Excel- <https://github.com/kuczera/GraphReader/raw/master/data/RegisterH4.xlsx> oder als OpenOffice-Datei <https://github.com/kuczera/GraphReader/raw/master/data/RegisterH4.ods> heruntergeladen werden. In dem Tabellendokument befinden sich insgesamt drei Tabellen. In der Tabelle **Personen** sind die Einträge des Personenregisters aufgelistet und in der Tabelle **Orte** befindet sich die Liste aller Einträge des Ortsregisters. Schließlich enthält die Tabelle **APPEARS_IN** Information dazu, welche Personen oder Orte in welchen Regesten genannt sind. Der folgende Cypher-Query importiert die Einträge der Personentabelle in die Graphdatenbank und erstellt für jeden Eintrag einen Knoten vom Typ :IndexPerson:

⁴Vgl. die Vorbemerkung zum Register in Böhmer, J. F., *Regesta Imperii III. Salisches Haus 1024-1125. Tl. 2: 1056-1125. 3. Abt.: Die Regesten des Kaiserreichs unter Heinrich IV. 1056 (1050) - 1106. 5. Lief.: Die Regesten Rudolfs von Rheinfelden, Hermanns von Salm und Konrads (III.). Verzeichnisse, Register, Addenda und Corrigenda, bearbeitet von Lubich, Gerhard unter Mitwirkung von Junker, Cathrin; Klocke, Lisa und Keller, Markus - Köln (u.a.) (2018), S. 291.*

```
// Registereinträge Personen erstellen
LOAD CSV WITH HEADERS FROM "https://github.com/kuczera/GraphReader/raw/master/data/RegisterH4-Place-APPEARS_IN.csv"
AS line
CREATE (:IndexPerson {registerId:line.ID, name1:line.name1});
```

Mit dem folgenden Cypher-Query werden nach dem gleichen Muster aus der Tabelle Orte die Ortseinträge in die Graphdatenbank importiert.

```
// Registereinträge Orte erstellen
LOAD CSV WITH HEADERS FROM "https://github.com/kuczera/GraphReader/raw/master/data/RegisterH4-Place-APPEARS_IN.csv"
AS line
CREATE (:IndexPlace {registerId:line.ID, name1:line.name1});
```

Die beiden Befehle laden die CSV-Daten und übergeben die Daten zeilenweise an die weiteren Befehle (Hier an den MATCH- und den CREATE-Befehl). Im nächsten Schritt werden nun mit den Daten der APPEARS_IN-Tabelle die Verknüpfungen zwischen den Registereinträgen und den Regesten erstellt.

```
// PLACE_IN-Kanten für Orte erstellen
LOAD CSV WITH HEADERS FROM "RegisterH4-Place-APPEARS_IN.csv"
AS line
MATCH (from:IndexPlace {registerId:line.ID})
MATCH (to:Regesta {regnum:line.regnum2})
CREATE (from)-[:PLACE_IN {regnum:line.regnum,
    name1:line.name1, name2:line.name2}]->(to);
```

Mit zwei MATCH-Befehlen werden jeweils das Regest und der Registereintrag aufgerufen und mit dem CREATE-Befehl eine PLACE_IN-Kante zwischen den beiden Knoten angelegt, die als Attribute den Inhalt der Spalten name1 und name2 erhält. Analog werden die Verknüpfungen zwischen Regestenknoten und Personenknoten angelegt:

```
// PERSON_IN-Kanten für Person erstellen
LOAD CSV WITH HEADERS FROM "RegisterH4-Person-APPEARS_IN.csv"
AS line
MATCH (from:IndexPerson {registerId:line.ID}),
(to:Regesta {regnum:line.regnum2})
CREATE (from)-[:PERSON_IN {regnum:line.regnum, name1:line.name1,
    name2:line.name2}]->(to);
```

4.5 Exkurs 2: Die Hierarchie des Registers der Regesten Kaiser Friedrichs III.

In anderen Registern der Regesta Imperii, wie beispielsweise den Regesten Kaiser Friedrichs III., sind teilweise fünf oder mehr Hierarchiestufen vorhanden, die jeweils auch Entitäten repräsentieren (vgl. Abb. 4.8).

- Bamberg (Bayern), Stadt
- Bürger und Einwohner s. Herr
 - Bischof 178
 - – Anton (von Rotenhan) (1431-1459) **29, 35, 45, 46, 177, 216, 238, 564**
 - Domkapitel **35**
 - – Domdekan s. Rotenhan
 - – – Gericht des Domdekans 29
 - – Dompropst **375, 376, 377**
 - – – Georg von Schaumberg **45, 46, 410, 411-413, 426, 430**
 - – – Lehengericht des Dompropstes **410-413, 426, 430**
 - – – – Lehenrichter s. Truchseß
 - Tag **20, 30, 31, 102**

Abbildung 4.8: Ausschnitt aus dem Register des Heftes 19 der Regesten Kaiser Friedrichs III.

In diesen Fällen müssen die Hierarchien auch in der Graphdatenbank abgebildet werden, was durch zusätzliche Verweise auf die ggf. vorhandenen übergeordneten Registereinträge möglich wird.

nodeID	xmlID	topnodeID	name1	name3
1	A00000001		Aa, Johann von	Aa, Johann von ~
2	A00000002	1	Sophie von ~, Tc	Aa, Johann von ~ // Sophie von ~, Tochter Johanns, Bürgerin zu Köln
3	A00000003		Aach	Aach (Fluß durch Aach, n. Singen, Baden-Württemberg)
4	A00000004		Aach	Aach (n. Singen, Baden-Württemberg), Stadt
5	A00000005		Aache s. Aacher	Aache s. Aachen
6	A00000006		Aachen	Aachen (Aache; Nordrhein-Westfalen), Stadt
7	A00000007	6	Einwohner und B	Aachen (Aache; Nordrhein-Westfalen), Stadt // Einwohner und Bürger ; s. Col
8	A00000008	6	Fischmarkt	Aachen (Aache; Nordrhein-Westfalen), Stadt // Fischmarkt (Parwisch)
9	A00000009	6	Gerichte	Aachen (Aache; Nordrhein-Westfalen), Stadt // Gerichte
10	A00000010	6	Kurgericht	Aachen (Aache; Nordrhein-Westfalen), Stadt // Gerichte // Kurgericht
11	A00000011	6	Schöffenstuhl, k	Aachen (Aache; Nordrhein-Westfalen), Stadt // Gerichte // Schöffenstuhl, kgl.
12	A00000012	6	Richter und Sch	Aachen (Aache; Nordrhein-Westfalen), Stadt // Gerichte // Schöffenstuhl, kgl.
13	A00000013	6	Schöffen	Aachen (Aache; Nordrhein-Westfalen), Stadt // Gerichte // Schöffenstuhl, kgl.
14	A00000014	6	„Grafschaften“	Aachen (Aache; Nordrhein-Westfalen), Stadt // „Grafschaften“
15	A00000015	6	„Grasgebot“	Aachen (Aache; Nordrhein-Westfalen), Stadt // „Grasgebot“
17	A00000017	6	Meierei	Aachen (Aache; Nordrhein-Westfalen), Stadt // Meierei
19	A00000019	6	Brothaus	Aachen (Aache; Nordrhein-Westfalen), Stadt // Meierei // Brothaus
21	A00000021	6	Gewandhaus	Aachen (Aache; Nordrhein-Westfalen), Stadt // Meierei // Gewandhaus
22	A00000022	6	Grashaus	Aachen (Aache; Nordrhein-Westfalen), Stadt // Meierei // Grashaus
23	A00000023	6	Plankenhaus	Aachen (Aache; Nordrhein-Westfalen), Stadt // Meierei // Plankenhaus
24	A00000024	6	Tuchhaus	Aachen (Aache; Nordrhein-Westfalen), Stadt // Meierei // Tuchhaus
25	A00000025	6	Haus zum Haner	Aachen (Aache; Nordrhein-Westfalen), Stadt // Meierei // Haus zum Haner (zu
26	A00000026	6	zo der Geiss	Aachen (Aache; Nordrhein-Westfalen), Stadt // Meierei // zo der Geiss
27	A00000027	6	Rentmeister	Aachen (Aache; Nordrhein-Westfalen), Stadt // Rentmeister
29	A00000029	6	„Stadtbücher“	Aachen (Aache; Nordrhein-Westfalen), Stadt // „Stadtbücher“
30	A00000030	6	Vogtei	Aachen (Aache; Nordrhein-Westfalen), Stadt // Vogtei

Abbildung 4.9: Ausschnitt der Entitätentabelle des Registers der Regesten Friedrichs III.

Im Tabellenausschnitt (Abb. 4.9) wird jedem Registereintrag in der ersten Spalte eine **nodeID** als eindeutige Kennung zugewiesen. Bei Registereinträgen, die kein Hauptlemma sind, enthält die dritte Spalte **topnodeID** den Verweis auf die eindeutige Kennung **nodeID** des übergeordneten Eintrages. Beim Import in die Graphdatenbank wird diese Hierarchie über CHILD_OF-Kanten abgebildet, die vom untergeordneten Eintrag auf das übergeordnete Lemma verweisen. Damit ist die komplette Registerhierarchie im Graphen abgebildet. In der Spalte **name1** ist das Lemma angegeben. In der Spalte **name3** ist zusätzlich zum Lemma noch der gesamte Pfad vom Hauptlemma bis zum Registereintrag, jeweils mit Doppelslashes (//) getrennt. Bei tiefer gestaffelten Registern ist teilweise ohne Kenntnis der übergeordneten Einträge eine eindeutige Identifizierung eines Eintrages nicht möglich. So wird in Zeile 17 der o.a. Abbildung allein mit der Angabe aus der Spalte **name1** nicht klar, um welche **Meierei** es sich handelt. Mit dem kompletten Pfad des Registereintrages in der Spalte **name3** wird dagegen deutlich, dass die Aachener **Meierei** gemeint ist.

4.6 Auswertungsperspektiven

4.6.1 Personennetzwerke in den Registern

4.6.1.1 Graf Robert II. von Flandern in seinem Netzwerk

Nach dem Import können nun die Online-Regesten und die Informationen aus den Registern der Regesten Kaiser Heinrichs IV. in einer Graphdatenbank aus einer Vernetzungsperspektive abgefragt werden.⁵

Ausgangspunkt ist der Registereintrag von [Graf Robert II. von Flandern](#). Diesen Knoten finden wir mit folgendem Query.

```
// Robert II. von Flandern
MATCH (n:IndexPerson) WHERE n.registerId = 'H4P01822'
RETURN *;
```

Mit einem Doppelklick auf den `IndexPerson`-Knoten öffnen sich alle `Regesta`-Knoten, in denen Robert genannt ist. Klickt man nun wiederum alle Regestenknoten doppelt an, sieht man alle Personen und Orte, mit denen Robert gemeinsam in den Regesten genannt ist.

Dies kann auch in einem Cypher-Query zusammengefasst werden.

```
// Robert II. von Flandern mit Netzwerk
MATCH (n:IndexPerson)-[:PERSON_IN]->
(r:Regesta)<-[PERSON_IN]-
(m:IndexPerson)
WHERE n.registerId = 'H4P01822'
RETURN *;
```

In Abb. 4.10 wird das Ergebnis dargestellt.

Hier wird der `MATCH`-Befehl um einen Pfad über `PERSON_IN`-Kanten zu `Regesta`-Knoten ergänzt, von denen jeweils eine `PERSON_IN`-Kante zu den anderen, in den Regesten genannten `IndexPerson`-Knoten führt.

Nimmt man noch eine weitere Ebene hinzu, wächst die Ergebnismenge stark an (Abb. 4.11). Der folgende Query kann daher je nach Rechnerleistung etwas länger dauern.

```
// Robert II. von Flandern mit Netzwerk und Herrscherhandeln (viel)
MATCH
(n1:IndexPerson)-[:PERSON_IN]->(r1:Regesta)<-[PERSON_IN]-
(n2:IndexPerson)-[:PERSON_IN]->(r2:Regesta)<-[PERSON_IN]-
(n3:IndexPerson)
WHERE n1.registerId = 'H4P01822'
```

⁵Die nun folgenden Abfragen sind zum Teil einer Präsentation entnommen, die für die Summerschool der [Digitalen Akademie](#) im Rahmen des [Mainzed](#) entwickelt wurden. Die Präsentation findet sich unter der URL <https://digitale-methodik.adwmainz.net/mod5/5c/slides/graphentechnologien/RI.html>.



Abbildung 4.10: Robert mit den Personen, mit denen er gemeinsam in Regesten genannt wird.

```
RETURN *;
```

4.6.1.2 Graf Robert II. von Flandern und Herzog Heinrich von Niederlothringen

In der Graphdatenbank ist es über die Exploration der Beziehungen einer Person hinaus möglich, explizit die Verbindungen von zwei Personen abzufragen. In unserem nächsten Beispiel suchen wir jene Regesten, in denen [Graf Robert II. von Flandern](#) und [Herzog Heinrich von Niederlothringen](#) gemeinsam genannt sind.

```
// Robert II. von Flandern und Herzog Heinrich von Niederlothringen mit Netzwerk
MATCH
(n:IndexPerson)-[:PERSON_IN]->
(r:Regesta)<-[ :PERSON_IN]-(m:IndexPerson)
WHERE n.registerId = 'H4P01822'
AND m.registerId = 'H4P00926'
RETURN *;
```

Es zeigt sich, dass Robert und Heinrich in einem Regest gemeinsam genannt sind (Abb. 4.12).

Und dieses [Regest](#) berichtet von der Unterwerfung Roberts unter Heinrich IV.⁶

Heinrich feiert das Fest der Apostel, wobei sich Graf Robert von Flandern im Beisein mehrerer Fürsten unterwirft, namentlich der Erzbischöfe Friedrich von Köln und Bruno von Trier, der Bischöfe Otbert von Lüttich, Burchard von Münster, Burchard von Utrecht, Herzog Heinrich von Niederlothringen sowie mehrerer Grafen.

Möglicherweise haben beide aber gemeinsame Bekannte, also Personen mit denen sowohl Heinrich als auch Robert in unterschiedlichen Regesten gemeinsam genannt sind. Hierfür wird der Cypher-Query um eine Ebene erweitert.

```
// Robert und Heinrich mit allen gemeinsamen Personen und Regesten
MATCH (n1:IndexPerson)
- [:PERSON_IN]->(r1:Regesta)<-[ :PERSON_IN]-
(n2:IndexPerson)-[:PERSON_IN]->(r2:Regesta)
<-[:PERSON_IN]-(n3:IndexPerson)
WHERE n1.registerId = 'H4P00926'
AND n3.registerId = 'H4P01822'
RETURN *;
```

Ein erster Blick auf das Ergebnis (Abb. 4.13) zeigt, dass Heinrich allgemein besser vernetzt ist. Für die weitere Analyse ihres Verhältnisses ist nun die Lektüre der angegebenen Regesten notwendig. Hierfür lässt sich das Ergebnis noch etwas weiter aufbereiten, indem die zwischen den Personen liegenden Regesten in KNOWS-Kanten umgewandelt werden, die als zusätzliche Information die Angaben zu den Regesten enthalten.

⁶Vgl. RI III,2,3 n. 1487.

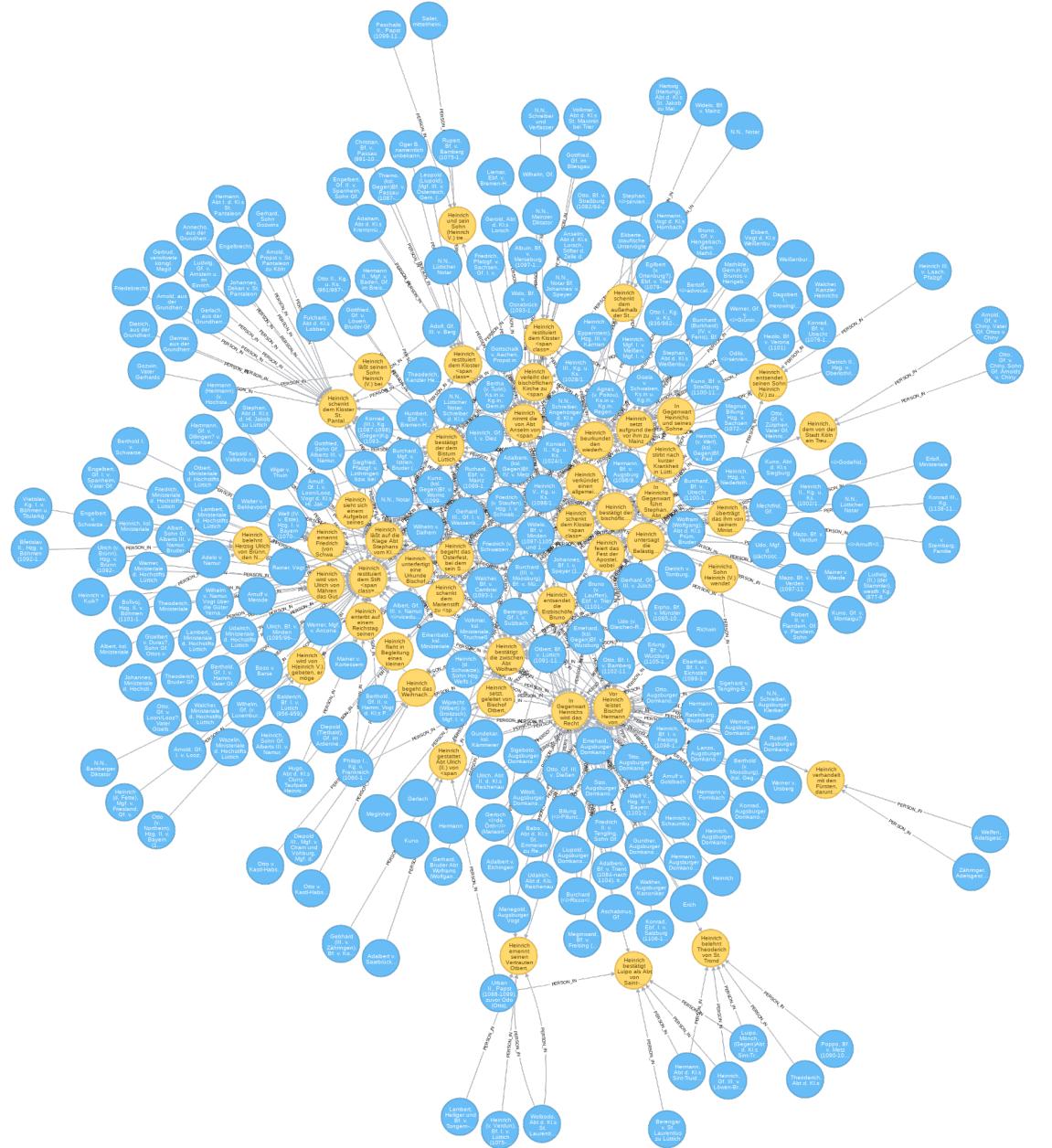


Abbildung 4.11: Robert mit Personen, die wiederum mit Personen gemeinsam in Regesten genannt sind.



Abbildung 4.12: Robert und Heinrich sind in einem Regest gemeinsam genannt.

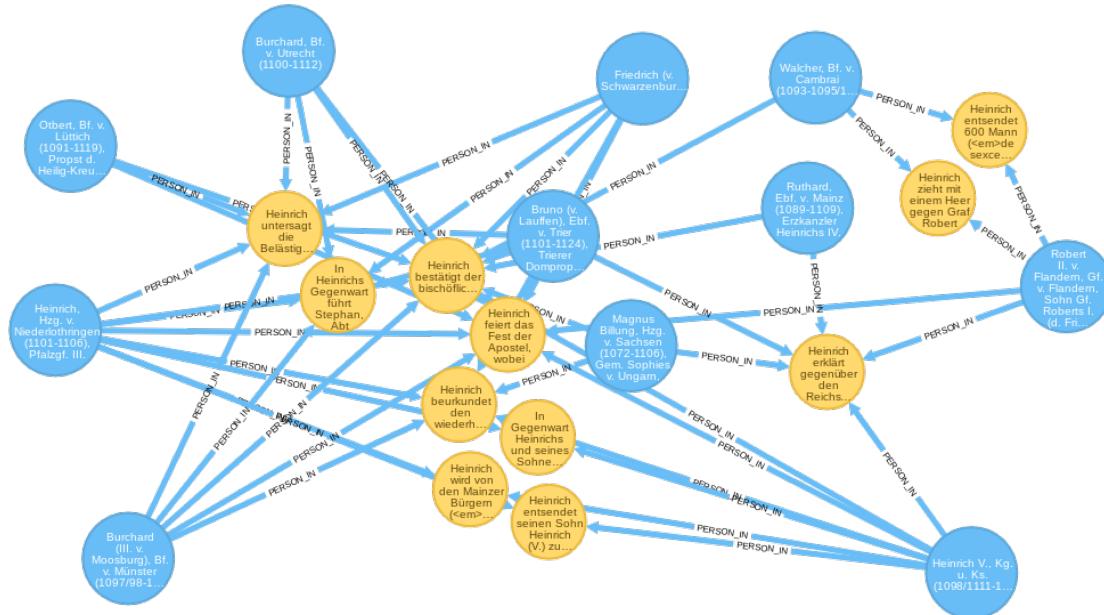


Abbildung 4.13: Robert und Heinrich mit den gemeinsamen Bekanntschaften.

```
// Rausrechnen der dazwischenliegenden Knoten
MATCH
  (startPerson:IndexPerson)-[:PERSON_IN]->
  (regest:Regesta)<-[:PERSON_IN]-(endPerson:IndexPerson)
WHERE startPerson.registerId in ['H4P01822', 'H4P00926']
WITH startPerson, endPerson, count(regest) as anzahl,
collect(regest.ident) as idents
CALL apoc.create.vRelationship(startPerson, "KNOWS",
{anzahl:anzahl, regesten:idents}, endPerson) YIELD rel
RETURN startPerson, endPerson, rel;
```

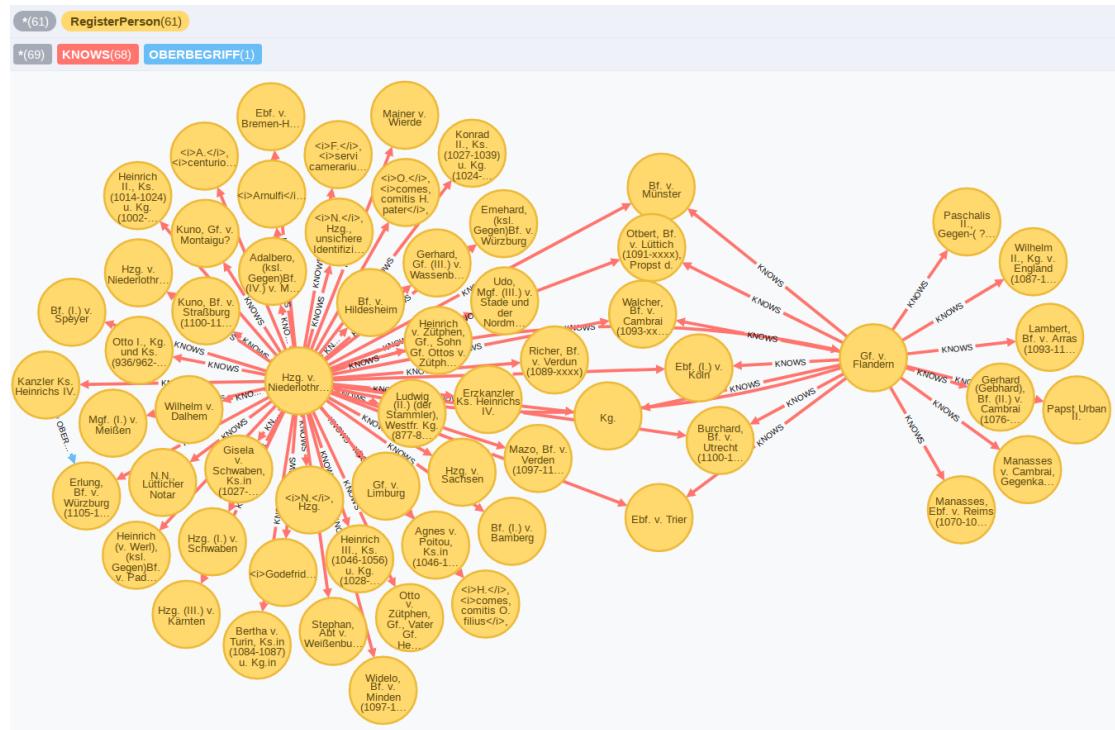


Abbildung 4.14: Robert und Heinrich mit den gemeinsamen Bekanntschaften.

In Abb. 4.14 sind die zwei Ego-Netzwerke von Heinrich (links) und Robert (rechts) mit den dazwischen liegenden gemeinsamen Bekanntschaften dargestellt. Es zeigt sich, dass Heinrich stärker sowohl mit Geistlichen als auch Weltlichen vernetzt war, während Robert insgesamt weniger Kontakte aber mit einem Schwerpunkt in der Geistlichkeit hatte.

Für den Historiker ist aber vor allem interessant, was in den Regesten steht, die Robert und Heinrich über die Mittelsmänner verbinden. Hierfür wird der Cypher-Query angepasst und sowohl Personen als auch die Regestentexte ausgegeben.

```
// Liste der Regesten als Ergebnis
MATCH
```

```

(startPerson:IndexPerson)-[:PERSON_IN]->
(regesta1:Regesta)<-[ :PERSON_IN ]-(middlePerson:IndexPerson)
-[:PERSON_IN]->(regesta2:Regesta)
<-[:PERSON_IN]-(endPerson:IndexPerson)
WHERE startPerson.registerId in ['H4P00926']
AND endPerson.registerId in ['H4P01822']
RETURN DISTINCT startPerson.name1,
regesta1.ident, regesta1.text,
middlePerson.name1, regesta2.ident,
regesta2.text, endPerson.name1;

```

In der folgenden Abbildung wird ein Ausschnitt der Ergebnistabelle gezeigt. In der ersten Spalte der Tabelle finden sich Robert, anschließend die Angaben zum Regest, mit dem er mit der mittleren Person (middlePerson.name1) verknüpft ist. Dem folgen schließlich die Angaben zum Regest, mit dem die mittlere Person mit Robert in der letzten Spalte verbunden ist. Die Tabelle in Abb. 4.15 bietet einen Überblick zur Überlieferungssituation aus der Perspektive der Regesta Imperii.

startPerson.name1	regesta1.ident	regesta1.text	middlePerson.name1	regesta2.ident	regesta2.text	endPerson.name1
"Heinrich, Hzg. v. Niederlothringen (1101-1106), Pfalzgf. III. v. Lothringen, Gf. v. Limburg"	"RI III,2,3 n. 1489"	"Heinrich bestätigt der bischöflichen Kirche zu Bamberg unter Bischof Otto zum Gedenken an seine Großeltern, Kaiser Konrad (II.) und Kaiserin Gisela, seine Eltern, Kaiser Heinrich (III.) und Agnes, seine Gemahlin, die Kaiserin Bertha, sowie besonders an seinen Verwandten, Kaiser Heinrich (II.), den Gründer der Bamberger Kirche, aufgrund der Intervention seines Sohnes, König Heinrichs V., der Erzbischöfe Friedrich von Köln, Bruno von Trier und Humbert von Bremen, der Bischöfe Otbert von	"Bruno (v. Lauffen), Ebf. v. Trier (1101-1124), Trierer Dompropst"	"RI III,2,3 n. 1487"	"Heinrich feiert das Fest der Apostel, wobei sich Graf Robert von Flandern im Beisein mehrerer Fürsten unterwirft, namentlich der Erzbischöfe Friedrich von Köln und Bruno von Trier, der Bischöfe Otbert von Lüttich, Burchard von	"Robert II. v. Flandern, Gf. v. Flandern, Sohn Gf. Roberts I. (d. von Flandern im Friesen) v. Flandern, Neffe Gf. Balduins VI. v. Flandern"

Abbildung 4.15: Robert und Heinrich mit den gemeinsamen Bekanntschaften.

4.6.2 Herrscherhandeln ausgezählt

Wie bereits oben erwähnt, wurde in einem ersten Test jeweils das erste Verb des Regesttextes extrahiert, lemmatisiert und in die Graphdatenbank eingespielt. Im folgenden werden nun einige Cypher-Querys vorgestellt, die dies beispielhaft auswerten.

```

// Herrscherhandeln ausgezählt
MATCH (n:Lemma)<-[:ACTION]->(m:Regesta)
RETURN n.lemma, count(h) as ANZAHL ORDER BY ANZAHL desc LIMIT 10;

```

n.lemma	ANZAHL
werden	145
schenken	133
bestätigen	109
begehen	95
verleihen	48
ernennen	36
nehmen	35
treffen	34
empfangen	29
erhalten	26

Die Ergebnisliste zeigt gleich die Einschränkungen, da das Hilfsverb *werden* aus dem textuellen Zusammenhang gerissen ist. Andererseits ergeben sich aber auch interessante Erkenntnisse zur Häufigkeitsverteilung von Herrscherhandeln in Regestentexten. Die Anwendung des Verfahrens auf Regestentexte ist dabei auf der einen Seite positiv, da bei der Erstellung der Regesten sehr stark auf formale Kriterien geachtet wird und so die Zusammenhänge gut zu erfassen sind. Auf der anderen Seite ist die Auswertung aber wiederum einen weiteren Schritt von der ursprünglichen Quelle entfernt.

4.6.3 Herrscherhandeln pro Ausstellungsort ausgezählt

Im folgenden Query kommt eine räumliche Komponente zur Abfrage hinzu, da das Lemma hier jeweils abhängig vom Ausstellungsort der Urkunde abgefragt wird.

```
// Herrscherhandeln pro Ausstellungsort
MATCH (n:Lemma)<-[h:ACTION]-(:Regesta)-[:PLACE_OF_ISSUE]->(p:Place)
WHERE p.normalizedGerman IS NOT NULL
RETURN p.normalizedGerman, n.lemma, count(h) as ANZAHL ORDER BY ANZAHL desc LIMIT 10;
```

p.normalizedGerman	n.lemma	ANZAHL
Mainz	begehen	15
Mainz	schenken	14
Goslar	schenken	13
Rom	werden	12
Regensburg	schenken	12
Goslar	begehen	11
Speyer	schenken	10
Worms	begehen	8
Regensburg	bestätigen	7
Regensburg	werden	7

In der ersten Spalte befindet sich der Ortsname, der aus der Property `normalizedGerman` des `Place`-Knotens stammt. In der zweiten Spalte wird das Lemma angegeben und in der dritten Spalte schließlich die Anzahl der jeweiligen Regesten. Interessant wäre hier auch noch die Ergänzung der zeitlichen Dimension, mit der dann der zeitliche Verlauf in die Auswertung miteinbezogen werden könnte.

4.6.4 Herrscherhandeln und Anwesenheit

Im nächsten Beispiel werden in einem Regest genannten Personen in die Auswertung des Herrscherhandelns mit einbezogen.

```
MATCH (p:IndexPerson)-[:PERSON_IN]-(r:Regesta)-[:ACTION]-(l:Lemma)
RETURN p.name1, l.lemma, count(l) AS Anzahl ORDER BY p.name1, Anzahl DESC;
```

p.name1	l.lemma	Anzahl
...
Adalbero, Metzer Domkanoniker, Kanzler Heinrichs IV., Kanzler (Gegen)Kg. Rudolfs v. Rheinfelden	schenken	21
Adalbero, Metzer Domkanoniker, Kanzler Heinrichs IV., Kanzler (Gegen)Kg. Rudolfs v. Rheinfelden	bestätigen	9
Adalbero, Metzer Domkanoniker, Kanzler Heinrichs IV., Kanzler (Gegen)Kg. Rudolfs v. Rheinfelden	verleihen	4
Adalbero, Metzer Domkanoniker, Kanzler Heinrichs IV., Kanzler (Gegen)Kg. Rudolfs v. Rheinfelden	erlassen	2
Adalbero, Metzer Domkanoniker, Kanzler Heinrichs IV., Kanzler (Gegen)Kg. Rudolfs v. Rheinfelden	übertragen	2
Adalbero, Metzer Domkanoniker, Kanzler Heinrichs IV., Kanzler (Gegen)Kg. Rudolfs v. Rheinfelden	ermäßigen	2
Adalbero, Metzer Domkanoniker, Kanzler Heinrichs IV., Kanzler (Gegen)Kg. Rudolfs v. Rheinfelden	gestatten	2
Adalbero, Metzer Domkanoniker, Kanzler Heinrichs IV., Kanzler (Gegen)Kg. Rudolfs v. Rheinfelden	vollziehen	1
Adalbero, Metzer Domkanoniker, Kanzler Heinrichs IV., Kanzler (Gegen)Kg. Rudolfs v. Rheinfelden	nehmen	1
Adalbero, Metzer Domkanoniker, Kanzler Heinrichs IV., Kanzler (Gegen)Kg. Rudolfs v. Rheinfelden	mindern	1
Adalbero, Metzer Domkanoniker, Kanzler Heinrichs IV., Kanzler (Gegen)Kg. Rudolfs v. Rheinfelden	setzen	1
...

Die Ergebnistabelle zeigt den Abschnitt zu Adalbero, einem Metzer Domkanoniker, mit der Häufigkeit des jeweiligen Herrscherhandeln-Lemmas.

4.6.5 Regesten 200 km rund um Augsburg

Mit dem folgenden Query werden für den Umkreis von 200 km rund um Augsburg alle Regesten aufgerufen.

```
// Entfernungen von Orten berechnen lassen
MATCH (n:Place)
WHERE n.normalizedGerman = 'Augsburg'
WITH n.latLong as point
MATCH (r:Regesta)
WHERE distance(r.latLong, point) < 200000
AND r.placeOfIssue IS NOT NULL
AND r.placeOfIssue <> 'Augsburg'
RETURN r.ident, r.placeOfIssue,
distance(r.latLong, point) AS Entfernung
ORDER BY Entfernung;
```

Solche Queries lassen sich auch mit zeitlichen Abfragen kombinieren und bieten sehr flexible Abfragemöglichkeiten.

4.6.6 Welche Literatur wird am meisten zitiert

Beim Import der Regesten in die Graphdatenbank werden die mit dem RI-Opac verlinkten Literaturtitel als eigenständige **Reference**-Knoten angelegt und jeweils mit dem **Regesta**-Knoten verknüpft. Diese Verknüpfung wird mit dem folgenden Query abgefragt, ausgezählt und aufgelistet.

```
MATCH (n:Reference)-[r:REFERENCES]-(m:Regesta)
RETURN n.title, count(r) AS Anzahl
ORDER BY Anzahl DESC LIMIT 10;
```

n.title	ANZAHL
Stumpf	215
Böhmer	201
Ldl	101
Jaffé	60
Schmale	56
Buchholz	51
Scheffer-Boichorst	50
Wauters	39
Dobenecker	33
Remling	28

Mit diesen Daten lassen sich Zitationsnetzwerke in den Regesten darstellen mit denen

Regesten gefunden werden können, die auf Grund der gemeinsam zitierten Literatur die gleichen inhaltlichen Schwerpunkte aufweisen können.

4.6.7 Der Import zusammengefasst

Den kompletten Cypher-Code für die Erstellung der Graphdatenbank ist zusammengefasst über eine Textdatei abrufbar. Es ist zu empfehlen, die aktuelle Version von neo4j-Desktop zu installieren, eine Graphdatenbank anzulegen und in der Graphdatenbank die Apoc-Bibliothek zu installieren. Inzwischen ist es möglich, in der Befehlszeile des neo4j-Browsers auch mehrere Befehle nacheinander ausführen zu lassen. Alternativ kann nach dem Start der Graphdatenbank im Reiter Terminal mit dem Befehl bin/cypher-shell die cypher-shell aufgerufen werden. In diese Shell werden dann alle Befehl gemeinsam hineinkopiert und ausgeführt. Alternativ zur Installation von neo4j kann auch auf den Internetseiten von neo4j eine Sandbox erstellt werden.

4.7 Zusammenfassung

In diesem Kapitel wurden die Schritte zum Import der Regesten Kaiser Heinrichs IV. in die Graphdatenbank neo4j erläutert sowie verschiedene Auswertungsbeispiele vorgestellt.

5 Verwandtschaft im Graphen

In diesem Kapitel wird am Beispiel eines Ausschnitts der Daten des Projekts **Nomen et Gens**¹ die Modellierung von Verwandtschaft in der Graphdatenbank neo4j dargestellt.²

5.1 Das Projekt Nomen et Gens

Das Projekt **Nomen et Gens** (NeG) zielt darauf ab, alle schriftlich belegten Namen und Personen Kontinentaleuropas in den vier Jahrhunderten vor Karl dem Großen (also von 400 bis 800 nach Christus) zu erfassen. Die Datenbank des Projekts geht auf ein erfolgreich abgeschlossenes DFG-Projekt zurück und wird aktuell von den Projektbeteiligten weiter betreut und sukzessive ausgebaut. Neben den Quellen der Personennennung, den unterschiedlichen Namensformen usw. werden auch die Verwandtschaftsbeziehungen zwischen identifizierten Personen in der Datenbank abgelegt. Dabei werden bis zu 16 verschiedene Verwandtschaftsbeziehungen in der Datenbank gespeichert, wie z. B. Bruder, Schwester, Sohn, Tochter, Vater, Mutter, Onkel oder Nichte. Bei einem Personendatensatz werden dann jeweils die Verwandtschaftsbeziehungen aufgelistet, so dass man sich ggf. jeweils von Person zu Person durchklicken muss, bis man am Ziel angelangt ist.

5.2 Nomen et Gens im Graphen

Vor diesem Hintergrund bot es sich an, die Personen und die zugehörigen Verwandtschaftsbeziehungen in die Graphdatenbank neo4j zu transferieren und anschließend graphbasierte Abfrageperspektiven zu testen.

Abb. 5.1 zeigt die ersten Ergebnisse des Datenbankimports. Aus der Visualisierung werden die zahlreichen redundanten Beziehungen deutlich, die in der Folge zu neuen Modellierungsansätzen für die Verwandtschaftsbeziehungen führten. Ergebnis der Überlegungen war die Reduzierung der möglichen Verwandtschaftsbeziehungen auf die

¹Informationen zum Projekt „Nomen et Gens“ finden Sie unter <http://www.neg.uni-tuebingen.de/> (abgerufen am 10.08.2018).

²Dieses Kapitel geht in großen Teilen zurück auf meinem Aufsatz Graphentechnologien in den Digitalen Geisteswissenschaften, in: ABI Technik 2017; 37(3): 179–196, <https://doi.org/10.1515/abitech-2017-0042>. URL: <https://www.degruyter.com/downloadpdf/j/abitech.2017.37.issue-3/abitech-2017-0042/abitech-2017-0042.pdf>, insbesondere die Seiten 179 bis 182 und wurde nur geringfügig ergänzt. Herrn Prof. Dr. Steffen Patzold danke ich herzlich für die Erlaubnis, die Nomen et Gens Daten im Rahmen dieser Publikation zu verwenden (mit Mail vom 22.01.2019).

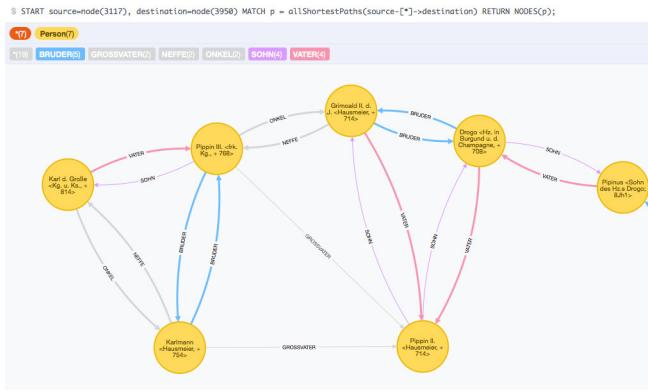


Abbildung 5.1: Erste Importergebnisse

zwei Kantentypen KIND und VERHEIRATET_MIT. Dabei wird eine Kante vom Typ KIND für eine Elternteil-Kind-Beziehung nur einmal vergeben, während eine Kante vom Typ VERHEIRATET_MIT immer zweifach in jeweils umgekehrter Richtung angelegt wird. Dies ergibt sich aus der Überlegung, dass eine Elternteil-Kind-Beziehung gerichtet ist und zwar in unserem Fall vom Elternteil zum Kind hin, während eine VERHEIRATET_MIT-Beziehung ungerichtet ist: Wenn eine Person mit einer anderen Person verheiratet ist, ist die andere Personen automatisch auch mit der ersten verheiratet. Da im Property-Graph-Modell von neo4j jede Kante genau eine Richtung haben muss, wird die VERHEIRATET_MIT-Kante zweimal in jeweils unterschiedliche Richtung angelegt, während bei der hierarchischen Elternteil-Kind-Beziehung eine Kante ausreicht (vgl. Abb 5.2).

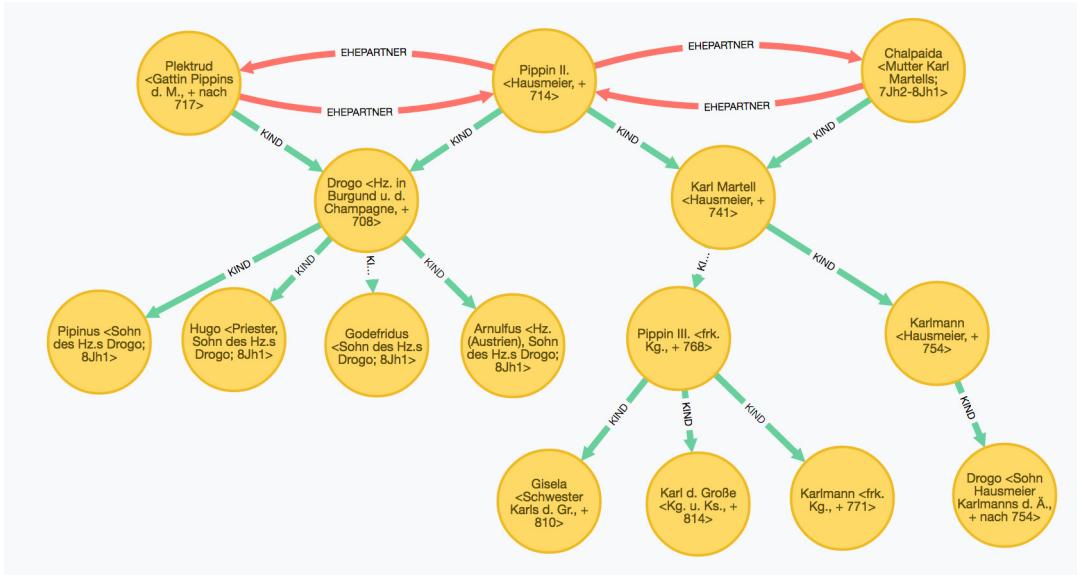


Abbildung 5.2: Die Urenkel Pippins

5.3 Sind Berchar und Karl der Große verwandt ?

Im folgenden Beispiel soll das Potential der Graphmodellierung von Verwandtschaftsbeziehungen demonstriert werden. In der Datenbank gibt es die Person Berchar (Abb. 5.3). Berchar war ein Hausmeier König Theuderichs III. Die Frage ist nun, ob dieser Berchar mit Karl dem Großen verwandt ist. In der NeG-Datenbank ist ein Verwandtschaftsverhältnis von Berchar zu Karl dem Großen nicht direkt ableitbar.

Prosopographisches			ID: P7119
Person	Berchar <Hausmeier Kg. Theuderichs III., + um 688/90>		
andere Namen	Berhar		
Geschlecht	m		
Kommentar	Berchar folgte um 686 seinem Schwiegervater Waratto als neustrischer Hausmeier nach. Er war im Konflikt mit Pippin II. d. Mittleren und wurde im Zuge dieser Auseinandersetzungen um 690 (oder 688) erschlagen (Berchar, LMA 1, 1980, 1931).		
Personenkommentar			
Stand	vir inluster		
Ämter	Amt maior domus consiliarius	Zeitraum 686-690 687	
Ethnie	Ethnie	Typus der Zuschreibung	
Verwandte	Name d. Person Drogo Waratto Anstrud Adaltrud Ansflid	Verwandtschaftsgrade Schwiegersonn Schwiegervater Ehepartner Tochter Schwiegermutter	

Abbildung 5.3: Berchar in der Nomen-et-Gens-Datenbank

In der Graphdatenbank neo4j wird für eine solche Fragestellung eine Shortest_Path-Abfrage verwendet, die den kürzesten möglichen Weg zwischen zwei Knoten zurück liefert, sofern es einen gibt. Der folgende Cypher-Befehl liefert den Pfad zwischen dem Personenknoten Karls des Großen mit der NeG-ID 7404 und dem Personenknoten von Berchar mit der NeG-ID 7119. Dabei wird die Länge des abzufragenden Pfades auf 15 Kanten begrenzt.

```
// shortest_path-Abfrage von Karl dem Großen zu Berchar
MATCH (KdG:Person { nid:'7404' })
MATCH (Berchar:Person { nid:'7119' })
p = shortestPath((KdG)-[*..15]-(Berchar))
RETURN p;
```

Das Ergebnis (Abb. 5.4) zeigt, dass Berchar tatsächlich mit Karl dem Großen verwandt ist. Er ist nämlich der Schwiegervater von Drogo (Herzog in Burgund und der Champagne, gest. 708), der wiederum der Bruder des Großvaters Karls des Großen ist.

5.4 Zusammenfassung

Mit diesem Beispiel sind die interessanten Erschließungs- und Modellierungsperspektiven für die digitale Genealogie nur angedeutet. Mit Graphentechnologien lässt sich

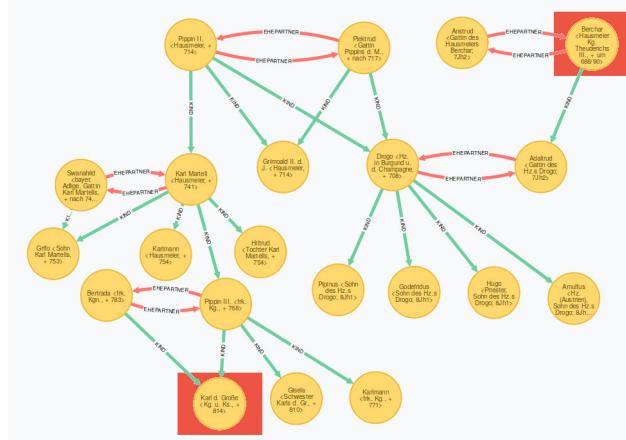


Abbildung 5.4: Der kürzeste Pfad (shortestPath) von Karl zu Berchar.

intuitive Datenmodellierung mit sehr flexiblen Erschließungs- und Abfragemöglichkeiten kombinieren.

6 Zusammenfassung

Anhand der verschiedenen Beispiele konnte gezeigt werden, dass Graphentechnologien hervorragend für die Modellierung, Speicherung und Analyse hochvernetzter Daten geeignet sind. Ebenso konnten die Beispiele zeigen, dass die digitalen Geisteswissenschaften reich an hochvernetzten Daten sind. Gleichzeitig lassen sich mit dem einer Mind-Map sehr ähnlichen Modell Forschungsdaten und Forschungsfragestellungen tatsächlich in einer Weise modellieren, die dem menschlichen Denken sehr nahe kommt. Damit können Graphentechnologien gleichsam als Brücke zwischen den geisteswissenschaftlichen Fachdisziplinen und den informationstechnologischen Herausforderungen und Perspektiven des digitalen Zeitalters dienen.

So gelingt es in den digitalen Geisteswissenschaften mit dem Graphenmodell bei der Modellierung und Strukturierung von Forschungsdaten und Forschungsfragestellungen die Kluft zwischen Informatiker und Geisteswissenschaftler zu schließen, da der Graph eine für beide Seiten verständliche Plattform bietet. Für den Informatiker ist er hinreichend genau und berechenbar und für den Geisteswissenschaftler wegen seiner Schema- und Hierarchiefreiheit ausreichend flexibel. Gerade diese Eigenschaften, mit denen sich die beiden zentralen Zweige der Digitalen Geisteswissenschaften vereinen lassen, machen Graphen zu einem Schlüsselkonzept der Geisteswissenschaften des 21. Jahrhunderts.