Graphbasierte digitale Editionen

VON ANDREAS KUCZERA VERÖFFENTLICHT 2016-04-19

Das Problem

Bei digitalen Editionen in den Geisteswissenschaften ist XML heute ein Standard. Oft wird TEI-XML für die Auszeichung literarischer und historischer Quellen verwendet.
Wenn nun in einem Dokument multiple Hierarchien ausgezeichnet werden sollen, ergeben sich Probleme. Hierzu gehört die fehlende Möglichkeit, sich überlappende Analyseebenen auszuzeichnen.
Ein einfaches und leicht nachvollziehbares Beispiel sind sich überlappende Hierarchien. Möchte man die inhaltliche Struktur des Textes und die formale Struktur des Texträgers in einem Dokument auszeichnen, stößt man auf das Problem der Überlappung z.B. bei Seiten und Kapiteln. Die TEI versucht dies zu lösen, indem sie sich für den Inhalt als primäre Struktur entschieden hat (und <p>) und formale Informationen mit leeren Elementen tagged (<lb> und <pb>).
Ein anderes Beispiel sind verschiedene Versionen eines Textes in einem XML-Dokument oder, wenn verschiedene Nutzerannotationen auf dem Text in einem Dokument ausgezeichnet werden sollen. Dann hat man schnell sich überlappende Strukturen auf einer Ebene, die nicht in einzelne Hierarchien unterteilt werden können.
Zur Lösung dieser Problematik wurden verschiedene Formate entwickelt.1 Zum einen sah man einen Ansatz darin, mit verschiedenen Hilfskonstruktionen die Repräsentation mehrerer Hierarchien in normalem Inline-XML mit Hilfe von „Milestones“2 oder „Fragmentierung“ oder einer Mischung von beidem oder auch durch multiple Elemente zu realisieren. Allen Lösungsansätzen gemein ist eine größere Komplexität des XML-Codes bis hin zur totalen Unlesbarkeit.
In einem zweiten Ansatz wurde versucht, mit Standoff-Markup den Befund, auch als Primär-Datum bezeichnet, zu indizieren und die Erschließungsinformationen jeweils an die Indizes zu knüpfen. Mit diesem Ansatz ist die gleichzeitige Darstellung verschiedener Erschließungshierarchien an einem Befund möglich. Nachteilig bei dieser Lösung ist die Starrheit der Indizierung. Sollte sich nach Erschließung des Materials die Notwendigkeit ergeben, die Reihenfolge im Befund und damit die Indizierung zu ändern, müsste die gesamte Erschließung neu indiziert werden. Momentan ist kein Editionstool in der Lage, diese Aufgabe unter gleichzeitiger Berücksichtigung von Usability-Gesichtspunkten zu lösen.

Der Lösungsansatz

In diesem Artikel soll ein neuer Ansatz für die Modellierung digitaler Editionen vorgestellt werden, mit dem der Befund einer Edition codiert und gleichzeitig transparent mehrere Deutungshierarchien erstellt, durchsucht, exportiert, bearbeitet und langzeitarchiviert werden können. Die technische Grundlage bildet eine Graphdatenbank, in diesem Fall Neo4j.3
In dieser Graphdatenbank wird in einem ersten Schritt der Befund, also der zu edierende Text als Graph angelegt. In einem zweiten Schritt werden anschließend Erschließungsinformationen ergänzt.
Zur Erläuterung der Grundzüge des technischen Modells wird ein Auszug aus dem Entwurf eines Regests Kaiser Friedrichs III. gezeigt. Der Text des Regestenentwurfs lautet:
Kaiser Friedrich III. bekräftigt die am 23. Juni 1464 erlassene Gerichtsordnung 1 und weitet ihre Gültigkeit 2 ausdrücklich auf Geistliche, in was wesens oder wirdigkeit und jüdische Personen aus. Er setzt fest, dass Geistliche vor dem Abt von Sankt Egidien oder dem Pfarrer von Sankt Sebald oder Sankt Lorenz einen Eid und jüdische Personen iren judisthen aid nach gepur ihres stats leisten sollen, dem zufolge sie nur vor einem Reichsgericht appellieren dürfen und das Appellationsurteil anerkennen müssen. Zuwiderhandlungen belegt er ausdrücklich auch für Geistliche und jüdische Personen mit einer je zur Hälfte der ksl. Kammer und den Nürnbergern zufallenden Pön von 100 Mark lothigen Goldes. am sambstag vor sant Johannis tag zu sunwenden3.

1Vgl. Reg. Nr. (475a).

2Sachhinweis: Ausweitung der Nürnberger Gerichtsordnung.

3am sampstag nach sandt Johanns tag zu sunbennden in 483.

In einem ersten Schritt wird nun der Regestentext selbst in die Graphdatenbank eingespielt. Hierfür wird eine Liste der Worte in ihrer vorkommenden Reihenfolge erstellt und diese dann in die Graphdatenbank Neo4j importiert.
Hier ein Ausschnitt aus der Liste der Befehle, mit denen für jedes Wort jeweils ein Knoten erstellt wird:

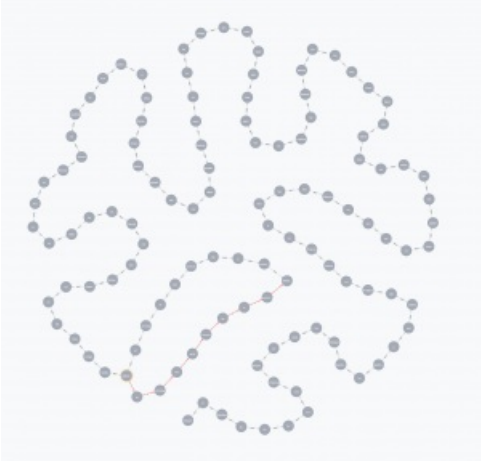
```
CREATE (:Wort {w_id:1, Wort:'Friedrich'});
CREATE (:Wort {w_id:2, Wort:'III.'});
CREATE (:Wort {w_id:3, Wort:'bekräftigt'});
CREATE (:Wort {w_id:4, Wort:'die'});
CREATE (:Wort {w_id:5, Wort:'am'});
CREATE (:Wort {w_id:6, Wort:'23.'});
CREATE (:Wort {w_id:7, Wort:'Juni'});
CREATE (:Wort {w_id:8, Wort:'1464'});
```

```
CREATE (:Wort {w_id:9, Wort:'erlassene'});
CREATE (:Wort {w_id:10, Wort:'Gerichtsordnung'});
CREATE (:Wort {w_id:11, Wort:'und'});
CREATE (:Wort {w_id:12, Wort:'weitet'});
…
```

Die Kanten (Verknüpfungen) zwischen den Knoten (also den Wörtern) werden mit folgenden Befehlen erstellt:

```
MATCH (from {w_id:1}), (to {w_id:2}) CREATE from-[:NEXT]->to;
MATCH (from {w_id:2}), (to {w_id:3}) CREATE from-[:NEXT]->to;
MATCH (from {w_id:3}), (to {w_id:4}) CREATE from-[:NEXT]->to;
MATCH (from {w_id:4}), (to {w_id:5}) CREATE from-[:NEXT]->to;
MATCH (from {w_id:5}), (to {w_id:6}) CREATE from-[:NEXT]->to;
MATCH (from {w_id:6}), (to {w_id:7}) CREATE from-[:NEXT]->to;
MATCH (from {w_id:7}), (to {w_id:8}) CREATE from-[:NEXT]->to;
MATCH (from {w_id:8}), (to {w_id:9}) CREATE from-[:NEXT]->to;
MATCH (from {w_id:9}), (to {w_id:10}) CREATE from-[:NEXT]->to;
MATCH (from {w_id:10}), (to {w_id:11}) CREATE from-[:NEXT]->to;
MATCH (from {w_id:11}), (to {w_id:12}) CREATE from-[:NEXT]->to;
…
```

Damit ist der Regestentext in der Graphdatenbank abgebildet. Jeweils ein Knoten repräsentiert ein Wort und die Kanten ergeben den Textfluss (vgl. Abbildung).



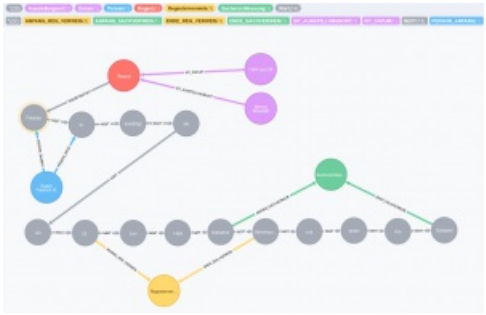
Das Regest beginnt unten und läuft dann gegen den Uhrzeigersinn rundherum. Am Ende spaltet sich die Textkette auf und zeigt eine Variante im Text. Bei der Erstellung des Regests hatte sich nämlich gezeigt, dass die beiden dem Regest zu Grunde liegenden Urkunden unterschiedliche Ausstellungsdaten haben (vgl. auch die oben beim Regestentext angezeigte Fußnote 2). Diesem wurde hier durch die Aufspaltung der Textkette in eine weiterlaufende Textkette und eine parallel dazu verlaufende Versionskette Rechnung getragen. Hierzu aber später noch mehr.

Nun fügen wir zum Text die erschließenden Informationen hinzu:

```
// Regestenknoten erstellen
CREATE (:Regest{reg_id:1, name:'Regest', uri:'http://www.regesta-imperii.de/id/1464-04-23_1_0_13_0_0_4066_4067'});
MATCH (from{reg_id:1}),(to{w_id:1}) CREATE from-[:REGESTENTEXT]->to;
// Ausstellungsort
CREATE (:Ausstellungsort{ao_id:1,name:'Wiener Neustadt', long:'16.25′, lat:'47.816667′, getty_id:'7003181'});
MATCH (from{ao_id:1}),(to{reg_id:1}) CREATE from-[:IST_AUSSTELLUNGSORT]->to;
// Datum
CREATE (:Datum{datum_id:1,Datum:'1464 Juni 30'});
MATCH (from{datum_id:1}),(to{reg_id:1}) CREATE from-[:IST_DATUM]->to;
// Regestenverweis erzeugen
CREATE (:Regestenverweis {f_id:1, name:'Regestenverweis', text:'Vgl. Reg. 475A'});
MATCH (from{w_id:6}),(to{f_id:1}) CREATE from-[:ANFANG_REG_VERWEIS]->to;
MATCH (from{w_id:10}),(to{f_id:1}) CREATE from-[:ENDE_REG_VERWEIS]->to;
// Sachverweis auf Ausweitung der Gerichtsordnung erzeugen
CREATE (:Sacherschliessung {s_id:1, name:'Sacherschliessung', text:'Erweiterung Gerichtsordnung'});
```

```
MATCH (from{w_id:9}),(to{s_id:1}) CREATE from-[:ANFANG_SACHVERWEIS]->to;
MATCH (from{w_id:14}),(to{s_id:1}) CREATE from-[:ENDE_SACHVERWEIS]->to;
// Person
CREATE (:Person{PERS_id:1,name:'Kaiser Friedrich III.',gnd:'http://d-nb.info/gnd/118535773'});
MATCH (from {PERS_id:1}),(to{w_id:1}) CREATE from-[:PERSON_ANFANG]->to;
MATCH (from {PERS_id:1}),(to{w_id:2}) CREATE from-[:PERSON_ENDE]->to;
```

Damit ergibt sich nun ein neues Bild unseres Graphen, von dem in der nächsten Abbildung ein Ausschnitt gezeigt wird.



Der das Regest darstellende Graph beginnt an dem roten Regestenknoten. 4 Vom Ausstellungsortsknoten führt eine Kante zum Regestenknoten und vom Datumsknoten ebenso. Mit dem Ausstellungsortsknoten wären dann auch alle weiteren Regesten mit dem gleichen Ausstellungsort verknüpft und ebenso mit dem Datumsknoten alle Regesten mit dem Datum 30. Juni 1464. Schließlich startet vom Regestenknoten noch die graue Textkette des Regestentextes, wobei jeder Knoten jeweils ein Wort repräsentiert, dass mit einer NEXT-Kante auf das folgende Wort verweist. Die zwei Knoten „Friedrich" und „III." sind über PERSON_ANFANG- und PERSON_ENDE-Kanten mit einem blauen Personenknoten verknüpft, der die Person Friedrichs III. repräsentiert und auf die GND etc. verlinkt. Der gelbe Knoten repräsentiert einen Verweis auf ein weiteres Regest, in welchem die erste Fassung der Gerichtsordnung regestiert worden ist, und verweist über die Kanten ANFANG_SACHVERWEIS und ENDE_SACHVERWEIS.
Am Ende des Regestentextes wird beispielhaft am Original-Ausstellungsdatum gezeigt, wie im Graphmodell Textvarianten codiert werden können5 Von der Urkunde, die dem Regest zugrunde liegt, gibt es eine Abschrift, in der das Ausstellungsdatum abweicht. Während das Ausstellungsdatum der Originalurkunde im Text, verknüpft durch NEXT-Kanten, durchläuft, wird das Ausstellungsdatum der zweiten Urkunde als Variante mit roten Kanten parallel dargestellt.
Dabei spaltet sich eine Variante vom Text ab und repräsentiert die alternative Version. Hierbei sind auch Verknüpfungen zwischen den einzelnen Knoten der Varianten denkbar, die beispielsweise Synonyme, ähnliche bzw. unterschiedliche Begriffe verknüpfen (vgl. folgende Abb.).

Mit dem folgenden Befehl erhält man den reinen Regestentext aus der Datenbank zurück:

```
// Nur Regestentext
match (:Regest{name:'Regest'})-[:REGESTENTEXT]->(s), textpath=(s)-[:NEXT*]->( e)
where not (e)-[:NEXT]->()
return reduce(s="", x in nodes(textpath) | s + x.Wort + " ")
```

Fazit

Insgesamt kann das hier dargestellte Beispiel die Vorteile und Potentiale von graphbasierten digitalen Editionen nur andeuten.
Ich denke aber gezeigt zu haben, dass sich mit dieser Technik
mehrere Varianten einer Quelle in einem Graph modellieren lassen,
mehrere Erschließungshierarchien in einem einzigen Graph gemeinsam erstellen und transparent voneinander unterscheiden lassen,
mehrere Erschließungshierarchien gemeinsam abfragen lassen,
nachträgliche Änderungen am Befund (also dem Primär-Datum) möglich sind, ohne dass die Erschließungsinformationen neu erstellt werden müssten.
Viele dieser Punkte lassen sich auch mit XML-Technologien oder Standoff-Markup lösen, jedoch ist dies oft mit großem Aufwand und

nur schwer lesbarem XML-Code verbunden. Gerade die meist noch vorhandene „Lesbarkeit“ war aber immer ein gewichtiger Grund für die Wahl von XML als (Standard-)Format. Da dies bei der heute notwendigen Komplexität der Auszeichnung nun nicht mehr gegeben ist, spricht aus meiner Sicht nichts dagegen, den Schritt zu graphbasierten digitalen Editionen zu gehen. Der Inhalt einer Graphdatenbank, also der Graph, ist einerseits menschenlesbar und kann andererseits sehr komplexe Erschließungshierarchien abbilden, ohne dass man sich um Auszeichnungsgrenzen Gedanken machen müsste. Für die Bedarfe der Langzeitarchivierung lassen sich graphbasierte digitale Editionen entweder als eine (falls es nur eine Erschließungshierarchie gibt) oder mehrere (wenn es mehrere Erschließungshierarchien gibt) XML-Dateien abspeichern und später wieder in eine Graphdatenbank einlesen. Auch eine Archivierung über Standoff-Markup ist denkbar.
Dies alles ist in einem Modell möglich, das dem menschlichen Denken und der Art, wie Menschen Informationen strukturieren und abspeichern, entgegen kommt. Graphbasierte digitale Editionen sind daher der nächste wichtige Schritt für die Digital Humanities, da mit ihnen den edierenden Disziplinen sehr flexible und einfach zu nutzende Editionswerkzeuge an die Hand gegeben und andererseits über graphbasierte Abfragesprachen wie z.B. Cypher diese komplexen Editions- und Erschließungsstrukturen abgefragt und erschlossen werden können.

DOWNLOAD
(PDF/A-Version)

Zitationsempfehlung/Suggested citation: Andreas Kuczera: Graphbasierte digitale Editionen, in: Mittelalter. Interdisziplinäre Forschung und Rezeptionsgeschichte, 19. April 2016, http://mittelalter.hypotheses.org/7994
Zu diesem Abschitt vgl. vor allem Jettka, Daniel (2011): Repräsentation, Verarbeitung und Visualisierung von multiplen Hierarchien mit XStandoff und XSLT. Masterarbeit zur Erlangung des akademischen Grades Master of Arts (M.A.) an der Fakultät für Linguistik und Literaturwissenschaft der Universität Bielefeld, Online-Version: http://www.daniel-jettka.de/pdf/Multiple_Hierarchien_mit_XStandoff_und_XSLT.pdf, abgerufen am 1.4.2016, der die Problematik überzeugend darlegt. [↩]
Vgl. hierzu http://www.tei-c.org/release/doc/tei-p5-doc/de/html/ref-milestone.html, abgerufen am 31.3.2016. [↩]
Vgl. www.neo4j.com. [↩]
Die Farben der Knoten und Kanten sind willkürlich gewählt und dienen hier vor allem der Übersichtlichkeit. [↩]
Im Originalregest ist die Angabe als Fußnote realisiert. [↩]