Leistungsfähige und einfache Suchen in lexikografischen Datennetzen Ein interaktivvisueller Query Builder für Property-Graphen

Meyer, Peter

meyer@ids-mannheim.de Institut für Deutsche Sprache, Deutschland

Einleitung: Property-Graphen für lexikografische Ressourcen

Klassische XML-basierte lexikografische Ressourcen können durch Graphenstrukturen mit zusätzlichen Vernetzungen und Informationen angereichert werden (M#chura 2016).1 Dabei werden die Artikel eines Wörterbuchs zunächst durch eigenständige XML-Dokumente repräsentiert; bestimmte XML-Elemente in diesen Dokumenten - die in typischen Anwendungsfällen z.B. den im Artikel gebuchten Wörtern oder deren Bedeutungsdefinitionen entsprechen - können dann zusätzlich in einer Graphdatenbank für Property-Graphen (vgl. Robinson / Eifrem / Webber 2013) durch Knoten verschiedener Typen repräsentiert werden. Im Redaktionsprozess können auch z.B. weitere Knoten hinzugefügt werden, um zusätzliche Information abzubilden. Kanten zwischen solchen Knoten können nicht nur bereits vorhandene relationale Informationen aus den Quelldokumenten, sondern auch zusätzliche, insbesondere auch dokumentübergreifende, Relationen zum Ausdruck bringen. Der resultierende Graph fungiert dann als ausdrucksstarke zusätzliche Navigations- und Repräsentationsebene.

Ein Query Builder für die Graphensuche

Lexikograf/innen ebenso wie Endnutzer/innen einer solchen Ressource benötigen eine Zugriffsstruktur, die Suchen nach komplexen Konstellationen in solchen Graphen ermöglicht. Für Graphendatenbanken stehen zahlreiche Abfragesprachen zur Verfügung, deren Verwendung jedoch sehr voraussetzungsreich ist. Die Entwicklung von interaktiv-visuellen Systemen zur endnutzerfreundlichen Graphenabfrage ist aktuelles

Forschungsgebiet (vgl. z.B. Bhowmick / Choi / Li 2018; Pienta / Navathe / Tamersoy / Tong / Endert / Chau 2016).

Als eine auf die Bedürfnisse der digitalen Lexikografie zugeschnittene, sich insbesondere auch an interessierte Endnutzer sowie Lexikografen ohne IT-Vorkenntnisse richtende Lösung präsentiert das Poster einen visuellen Query Builder, der von den Komplexitäten der in vielen gängigen Property-Graphendatenbanksystemen implementierten Open Source-Abfragesprache Apache TinkerPop Gremlin (Rodriguez 2015; tinkerpop.com) abstrahiert.² Das Poster illustriert die Verwendung des Systems anhand einer Datenbank zu lexikalischen Entlehnungen aus dem Deutschen in andere Sprachen, die die mitunter verwickelten Entlehnungswege von Wörtern als Pfade in einem Graphen abbildet.

Abfragen werden im Browser durch das visuelle Zusammenstellen eines Baumes von Abfragekomponenten erzeugt, die Eigenschaften von Knoten beschreiben. Die so erstellten Abfragen sind zu jedem Zeitpunkt semantisch konsistent. Nach jeder Änderung an der Abfrage wird diese serverseitig in einer für Administratoren frei konfigurierbaren Weise in eine Gremlin-Graphtraversierungsanweisung umgesetzt und die Suchresultate in Echtzeit zurückgegeben. Angesichts der Mächtigkeit von Gremlin und der Möglichkeit, Traversierungen mit beliebigen Seiteneffekten zu verknüpfen, ist die direkte, manuelle Eingabe von Gremlin-Anweisungen nur in einer separaten, für Administratoren bestimmten Konsole möglich.

Im allgemeinen Fall werden *n*-Tupel von Knoten gesucht, die bestimmte Attribute aufweisen und zwischen denen nutzerdefinierte Pfade bestehen sollen. Entsprechend werden die Resultate tabellarisch als sortierbare *n*-Tupel präsentiert.



Abbildung 1. Beispiel für eine Query Builder-Suchanfrage in einem lexikografischen Netzwerk für Entlehnungsbeziehungen: Suche polnische Substantive, die als Lehnwort ohne Genuswechsel ins Ukrainische gewandert sind, wobei das Lehnwort oder eine Ableitung dazu nicht vor 1950 belegt ist.

Komplexere Abfragen

Eine Relation zwischen zwei Knoten (z.B. direkte Kante mit einem bestimmten Attribut; ein Pfad mit maximal 3 Kanten; ein Pfad beliebiger Länge) wird in einer speziellen Abfragekomponente als "relationales Quasi-Attribut" eines der beiden Knoten eingegeben; die weiteren Eigenschaften des jeweils anderen Knoten erscheinen dann auf der hierarchisch nächsttieferen Ebene unterhalb dieser Abfragekomponente, wie aus Abb. 1 ersichtlich. Das Kombinieren von Suchkriterien durch eine Boolesche Abfragekomponente ist nicht nur für echte Knotenattribute, sondern auch für solche relationalen Quasiattribute erlaubt. So sind alternative oder verbotene Pfade beschreibbar, die in Gremlin als Sub-Traversierungsroutinen verarbeitet werden müssen und in einer rein graphischen visuellen Metapher nicht mehr ohne weiteres darstellbar wären. Referenzieren anderer Knoten ist über ein sich automatisch aktualisierendes Nummerierungsschema möglich, um Sachverhalte wie "Knoten B hat einen anderen Wert für Attribut X als Knoten A" oder auch Zyklen und andere nicht-baumartige Konstellationen im Graph auszudrücken.

Schon bei kleinen Graphen können komplexere Abfragen leicht zu nicht akzeptablen Suchlaufzeiten führen (vgl. Wood 2012; Bonifati / Fletcher / Voigt / Yakovets 2018), die über Zeitbeschränkungen in der Graphtraversierung gekappt werden müssen. Durch geeignete Maßnahmen kann in vielen Fällen die Existenz weiterer Suchergebnisse festgestellt und autorisierten Nutzern über eine Warteschlange die Möglichkeit gegeben werden, ihre Suchabfrage vollständig abarbeiten zu lassen.

Für jedes gefundene Knoten- n-Tupel kann ein sie enthaltender Ausschnitt (Subgraph) des Gesamtgraphen angezeigt und bei entsprechender Autorisierung von lexikografischen Bearbeitern in einem frei konfigurierbaren Editor visuell redigiert werden (vgl. Abb. 2).

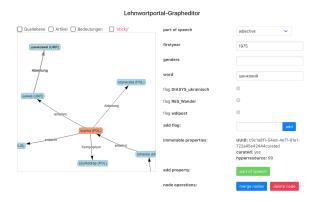


Abbildung 2. Suchergebnis mit passendem Ausschnitt aus dem Graphen und Editorfunktionalität.

Fußnoten

- 1. Die im Umfeld von Linked (Open) Data verwendeten Verfahren (vgl. Gracia / Kernerman / Bosque-Gil 2017) verwenden üblicherweise Graphendarstellungen lexikografischer Daten im RDF-Format, für die das hier vorgestellte, speziell für Property-Graphen und deren Abfragesprache Gremlin entwickelte Werkzeug nicht geeignet ist. Viele Arbeiten zu nutzerfreundlichen Suchwerkzeugen auf RDF-Netzen (z.B. Ferré 2017) sind jedoch für die hier behandelte Problematik sehr wohl von grundsätzlichem Interesse, weil sie in vergleichbarer Weise eine endnutzerfreundliche Zugriffsschicht über die RDF-Abfragesprache SPARQL legen.
- 2. Der Query Builder ist Komponente eines derzeit in Entwicklung befindlichen Open-Source-Softwaresystems zur Verwaltung und Online-Publikation graph-erweiterter lexikografischer Ressourcen (Meyer / Eppinger 2018), das im Rahmen des von der Fritz Thyssen Stiftung geförderten Projektes "Das Lehnwortportal Deutsch als Forschungs- und Publikationsplattform" entwickelt wird.

Bibliographie

Bhowmick, Sourav S. / Choi, Byron / Li, Chengkai (2018): Human Interaction with Graphs: A Visual Querying Perspective. San Rafael, CA: Morgan & Claypool Publishers.

Bonifati, Angela / Fletcher, George / Voigt, Hannes / Yakovets, Nikolay (2018): *Querying Graphs*. San Rafael, CA: Morgan & Claypool Publishers.

Ferré, Sébastien (2017): "Sparklis: An Expressive Query Builder for SPARQL Endpoints with Guidance in Natural Language", in: Semantic Web: Interoperability, Usability, Applicability 8(3): 405-418.

Gracia, Jorge / Kernerman, Ilan / Bosque-Gil, Julia (2017): "Toward Linked Data-Native Dictionaries", in: Kosem, Iztok / Tiberius, Carole / Jakubí#ek, Miloš / Kallas, Jelena / Krek, Simon / Baisa, Vít (eds.): Electronic lexicography in the 21st century. Proceedings of eLex 2017 conference. Brno: Lexical Computing 550-559 https://elex.link/elex2017/proceedings-download/ [letzter Zugriff 12. Oktober 2018].

M#chura, Michal (2016): "Data structures in lexicography: from trees to graphs", in: Horák, Aleš / Rychlý, Pavel / Rambousek, Adam (eds.): Proceedings of the Tenth Workshop on Recent Advances in Slavonic Natural Languages Processing, RASLAN 2016. Brno: Tribun EU 97-104.

Meyer, Peter / Eppinger, Mirjam (2018): "fLexiCoGraph: Creating and Managing Curated Graph-Based Lexicographical Data", in: #ibej, Jaka / Gorjanc, Vojko / Kosem, Iztok / Krek, Simon (eds.): Proceedings of the XVIII EURALEX International Congress. Lexicography in Global Contexts, 17-21 July, Ljubljana. Ljubljana: Znanstvena založba 1017-1022.

Pienta, Robert / Navathe, Shamkant / Tamersoy, Acar / Tong, Hanghang / Endert, Alex / Chau, Duen Horng (2016): "VISAGE: Interactive Visual Graph Querying", in: AVI: Proceedings of the Workshop on Advanced Visual Interfaces 272–279.

Robinson, Ian / Eifrem, Emil / Webber, Jim (2013): Graph Databases. Sebastopol, CA: O'Reilly & Associates. Rodriguez, Marko A. (2015): "The Gremlin Graph Traversal Machine and Language", in: Cheney, James / Neumann, Thomas (eds.): Proceedings of the 15th Symposium on Database Programming Languages (DBPL 2015). New York: The Association for Computing Machinery 1-10.

Wood, Peter T. (2012): "Query Languages for Graph Databases", in: SIGMOD Record 41(1): 50-60.