# GraphAnno

## Graphenbasierte Annotation

Christoph Rzymski, Volker Gast,
Lennart Bierkandt, Stephan Druskat

christoph.rzymski@uni-jena.de, volker.gast@uni-jena.de,
post@lennartbierkandt.de, stephan.druskat@hu-berlin.de

19. Januar 2017

# Inhalte

# Motivation und Hintergründe

## LinkType

‚Paradigmenneutrale' Annotation verschiedenster linguistischer Ebenen in einer Umgebung, einem Modell, einem Format.

## Software-Prototyping

GraphAnno[1] als Prototyp von Atomic[2]. Warum?

## Annotations-Prototyping

(Kleine, schnelle) Annotationsstudien in unrestringiertem Graphenmodell. Annotatoren können direkt mit der Arbeit beginnen.

---

[1] https://github.com/LBierkandt/graph-anno

[2] http://corpus-tools.org/atomic/

# Motivation und Hintergründe

(Linguistische) Annotationsprojekte sehen sich mit zwei elementaren Fragestellungen konfrontiert:

- Welches Annotationsschema soll verwendet werden? Die Antwort auf diese Frage ist durch verschiedene Einflüsse geprägt (Forschungsthema, Theorien, etc.).
- In welchem Datenformat sollen die Annotationen abgelegt werden?

Leider ist in der Praxis eine unabhängige Beantwortung dieser beiden Fragen häufig nicht möglich: Theorien diktieren Schemata und Schemata diktieren Datenformate.

„Können wir das überhaupt abbilden?“

# Motivation und Hintergründe

Annotations-Prototyping baut vor allem auf Geschwindigkeit: Schnelles Annotieren führt zu schnellem Feedback (‚validated learning‘).

Linguistische Daten sind Elemente, die in unterschiedlichen Beziehungen zueinander stehen. Es gilt also lediglich, die Elemente und deren Eigenschaften sowie die Beziehungen der Elemente und deren Eigenschaften zu modellieren.

Graphenbasierte Ansätze eignen sich hervorragend!

# Motivation und Hintergründe

Ursprüngliche Idee: Modellieren stark heterogener linguistischer Daten in unrestringierten Graphen. Ablage der Graphen in Neo4j[3]

Aktuelle Umsetzung: Neo4j bringt in der Einzelplatzanwendung gewisse Nachteile mit sich. Deswegen: Neo4j-kompatible Persistenz in JSON (Bray (2014), Bryan und Nottingham (2013)).

Zukunft: Weitere Pflege und Entwicklung sowie Übergang von Features auf Atomic.

[3] https://neo4j.com

# Technische Aspekte

Projektwebsite:

- https://github.com/LBierkandt/graph-anno

Pflege, Betreuung, Programmierung:

- Lennart Bierkandt[4]

Konzepte:

- unrestringierte, graphen-basierte Annotation
- kommandozeilengesteuerte Bedienung
- Einbettung in Browser

Persistenz:

- Neo4j-kompatibles JSON

[4] http://lennartbierkandt.de/

# Features und Einschränkungen

Features:

- schnelles, flüssiges Bedienkonzept
- visuell ‚freundliche' Mehrebenenannotation
- mächtige Abfragesprache, kompatibel zu freiem Eingabeformat
- flexibler Im- und Export

Einschränkungen:

- Heuristik zur Graphengenerierung nur bedingt konfigurierbar
- größere Datenmengen ($n > 10.000$) potenziell langsam, unübersichtlich

# Demonstration

Korpusdaten:

- Skopus: Manshadi, Allen und Swift (2011)
- Idi: Evans u. a. (forthcoming)

# Fragen

# Bibliographie


Bray, T. (2014). *The JavaScript Object Notation (JSON) Data Interchange Format*. RFC 7159. RFC Editor, S. 1–16. URL: https://tools.ietf.org/html/rfc7159.

Bryan, P. und M. Nottingham (2013). *JavaScript Object Notation (JSON) Patch*. RFC 6902. RFC Editor, S. 1–18. URL: https://tools.ietf.org/html/rfc6902.

Evans, N. u. a. (im Erscheinen). “The languages of Southern Papua New Guinea”. In: *The Languages and Linguistics of New Guinea: A Comprehensive Guide*. Hrsg. von B. Palmer. Berlin: de Gruyter Mouton.

# Bibliographie

Manshadi, Mehdi, James Allen und Mary Swift (2011). “A Corpus of Scope-disambiguated English Text”. In: *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*. Association for Computational Linguistics, S. 141–146.