Social Network Analysis on StackOverflow

# Einleitung

StackOverflow ist Teil des StackExchange Netzwerks und bietet eine Plattform für Fragestellungen im Bereich Softwarentwicklung. Im Rahmen des Moduls «Social Network Analysis» der FHNW Technik führen wir eine Analyse von Fragen, Antworten und Tags von StackOverflow aus.

Wir entwickeln eine Anwendung zur automatischen Abfrage, Aufbereitung und Speicherung dieser Daten. Die Software bietet Exportfunktionen an, die es erlauben, die gesammelten Daten in geeigneter Form in SNA-Tools wie Gephi weiter zu analysieren.

Die zentralen Fragestellungen dieser Arbeit sind folgende:

* Was sind die populärsten Technologien?
* In welcher Community hat man die besten Antwortchancen (relativ zur Community Grösse)?
* Welche Technologien werden zu Arbeits- und welche zu Hobby-Zwecken verwendet?
* Welche Technologien sind am stärksten miteinander verbunden?
* Welche Technologien stehen eher allein?
* Was haben schnell beantwortete Fragen gemeinsam?

# Datenbeschaffung

Die benötigten Daten sind alle frei auf StackOverflow.com verfügbar. Wir können den StackExchange-weiten REST Endpoint verwenden, um die Daten im json Format zu scrapen – wir sind hier bezüglich der Datenmenge zwar auf 10’000 API Requests (300, wenn nicht registriert) pro Tag eingeschränkt, dies ist aber mehr als ausreichend für dieses Projekt. Pro Request bekommen wir bis zu 100 Datensätze.

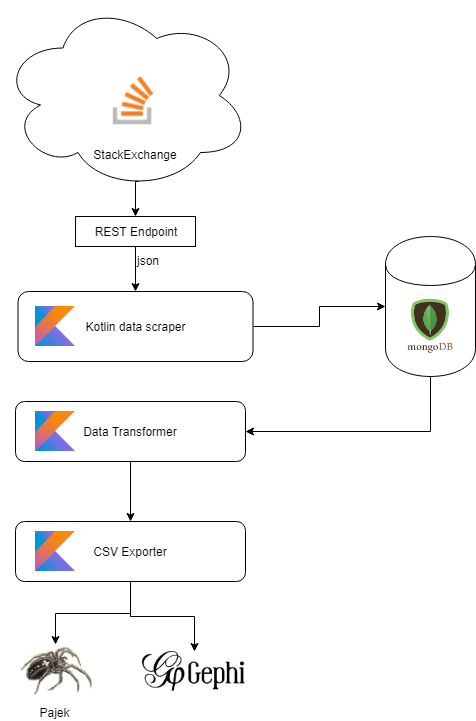
## API Einschränkungen

Eine Einschränkung ist, dass wir die Throttle Violation beachten. Wir dürfen nicht mehr als 300 (10'000 wenn registriert) tägliche und 30 sekündliche API Requests absenden. Zusätzlich muss der «backoff» Parameter respektiert werden, welcher bei einer Response mitgeliefert werden kann. All dies wurde bei der Implementierung unseres Scrapers beachtet und stellt keinerlei Probleme dar. Wir erhalten pro Request bis zu 100 Datensätze, können also täglich bis zu einer Million Datensätze pro IP scrapen. Wir aggregieren die Datensätze zusätzlich in einer MongoDB und werden so definitiv genug Daten zur Verfügung haben.

Zusätzliche Infos hier: <https://api.stackexchange.com/docs/throttle>

## Softwarearchitektur & Datenspeicherung

Die verwendeten Technologien beschränken sich auf Kotlin und Spring Boot. Mit einem eigenen Websraper fragen wir die Daten im json Format vom REST Endpoint ab, mappen sie zu Mongo Documents und speichern sie in einer MongoDB zwischen. Von dort werden die Daten dann transformiert und zu einem Graph in CSV Format umgewandelt, der dann in Gephi importiert werden kann.



## How to & Setup

Unsere Software ist auf Github verfügbar: <https://github.com/patricsteiner/sna_soscraper>

Am einfachsten klont man das Projekt und importiert es in IntelliJ. Von dort wird gradle automatisch alle nötigen Schritte vornehmen, um die Software zum Laufen zu bringen. Im Readme findet man zusätzliche Informationen: <https://github.com/patricsteiner/sna_soscraper/blob/master/README.md>

Wir haben während mehreren Tagen Daten gesammelt und diese in einer MongoDB (gratis hosting auf mlab.com) gespeichert. In der Version, die wir zu diesem Dokument mitliefern, sind die nötigen Credentials bereits konfiguriert, die Applikation kann also as-is gestartet und verwendet werden. Man kann aber auch eine eigene MongoDB verwenden, um Daten zu speichern. Dazu muss einfach die Datei appliaction.properties entsprechen angepasst werden.

Wenn man nur eine kleinere Menge an Daten abfragen und gleich weiterverwenden möchte ist es einfacher, die Daten einfach temporär im Memory zu halten, anstatt zu persistieren. Wir bieten dazu entsprechende Repository Klassen an (InMemoryQuestionRepository und InMemoryTagRepository). Um diese zu verwenden, muss einfach die @Repository Annotation dort gesetzt und beim MongoRepository entfernt werden.

# Datenanalyse

Im Rahmen dieser Arbeit verwenden wir insgesamt 77587 Datensätze, die wir während mehreren Tagen gescraped haben. Im Schnitt hat jede Frage 2.92 Tags.

questionRepository.findAll().size 🡪 77587

questionRepository.findAll().map { it.tags.size }.average() 🡪 2.92

## Rohdaten

Wie die Rohdaten im Detail aussehen, kann der Klasse Question entnommen werden: <https://github.com/patricsteiner/sna_soscraper/blob/master/src/main/kotlin/ch/fhnw/sna/soscraper/domain/Question.kt>

Die für uns wichtigsten Attribute zu jeder Frage sind nachfolgend aufgelistet:

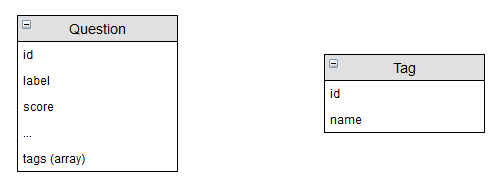
* Id
* Label
* Titel
* Erstellungsdatum
* Anzahl Antworten
* Hat akzeptierte Antwort
* Score
* Anzahl Views
* Tags

Die Daten kommen garantiert immer genau gleich an und es sind somit keine zwingenden Bereinigungsschritte nötig.

## Transformation

Aus den gesammelten Daten kreieren wir zwei verschiedene Sammlungen von Entitäten. Einerseits die Entität Question und andererseits die Entität Tag, wobei eine n-n Verknüpfung von Question zu Tag besteht. Es ist nötig, dass wir Tags (die einfach als String-Array einer Question geliefert werden) zu einer Entität mit eigener ID umwandeln, da das Tool Gephi eine id voraussetzt.

Wir erhalten also zwei Repositories mit folgenden Entitäten:



Aus diesen zwei Repositories können wir nun ein 2-Mode Netzwerk generieren. Jede Question und jedes Tag werden jeweils zu einem Knoten mit dem dazugehörigen Typ. Für jedes Tag einer Question wird eine Kante zwischen den entsprechenden Knoten generiert.

Für unsere Arbeit ist es jedoch nutzbringender, dieses 2-Mode Netzwerk (Question zu Tag) in ein 1-Mode Netzwerk (Tag zu Tag) zu transformieren. So können wir bei der Transformation noch nachfolgend beschriebene zusätzliche Informationen gewinnen. Diese Attribute ermöglichen uns erweiterte Analysen und werden uns helfen, die Kernfragestellungen dieser Arbeit zu beantworten.

* **Occurence**: Anzahl Vorkommen in allen Fragen
* **Views**: Summe der Views aller Fragen, die dieses Tag beinhalten
* **Answered**: Summe der beantworteten Fragen, die dieses Tag beinhalten
* **Unanswered**: Summe der unbeantworteten Fragen, die dieses Tag beinhalten
* **Bounty**: Summe der Bounties aller Fragen, die dieses Tag beinhalten
* **AnsweredRatio**: Verhältnis von beantworteten zu unbeantworteten Fragen, die dieses Tag beinhalten

# Soziale Netzwerkanalyse

Zur Analyse wurde der generierte ungerichtete Graph in Gephi importiert. Alle Mehrfachkanten wurden zusammengefügt und bekamen die Summe der Anzahl zusammengeführter Kanten als Gewicht.

## Kennzahlen zum Netzwerk

|  |  |  |
| --- | --- | --- |
| **Kennzahl** | **Wert** | **Kommentar** |
| Anzahl Knoten | 16’777 | Erstaunlich, dass es so viele Tags gibt! |
| Anzahl Kanten | 239’908 | Waren ursprünglich viel mehr, doppelte Kanten wurden zusammengefügt. |
| Average Degree | 3.484 | Jedes Tag ist im Schnitt mit so vielen anderen Tags direkt verbunden. |
| Average Weighted Degree | 37.299 | Diese Zahl sagt nicht viel aus, da die Streuung enorm gross ist (s. Abbildung 1) |
| Graph Density | 0 | Die Dichte ist sicher nicht 0, jedoch so klein, dass sie von Gephi als 0 ausgegeben wird. |

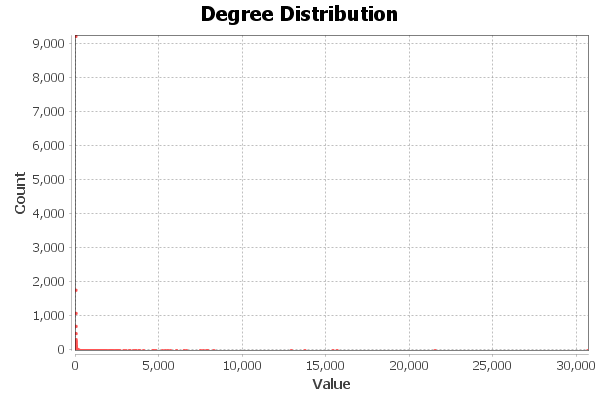


Abbildung 1

Weitere Metriken können in diesem riesigen Netzwerk nur mit enorm hohen Rechenaufwand berechnet werden, darum werden wir in den weiteren Analysen nur bestimmte Teilnetzwerke genauer anschauen.

## Other

# Ausblick

Was wäre Ihrer Meinung nach noch interessant zu analysieren, wenn das Projekt weitergeführt würde? Gäbe es noch weitere interessante Fragestellungen / weitere interes-sante Attribute, die eingesammelt oder sauber verarbeitet werden müssten? War die Daten-quelle passend oder sollte eine andere in Betracht gezogen werden? Falls ja, haben Sie eine gefunden?