How To Build A Knowledge Graph

Semantics Conference 2019, Tutorial
Elias Kärle & Umutcan Simsek,
STI2, University of Innsbruck, September 9th, 2019
About Us

Elias Kärle
PhD Student
elias.kaerle@sti2.at
Twitter: @eliaska

Umutcan Simsek
PhD Student
umutcan.simsek@sti2.at
Twitter: @umutsims
Acknowledgement

This tutorial is based on the work being done in the MindLab, an industrial research project for building knowledge graphs to be consumed by conversational agents in domains like tourism.

An extensive version of the content of this tutorial can be found in our upcoming book “Knowledge Graphs in Use” (working title)

https://mindlab.ai
About the Tutorial

The tutorial aims to introduce our take on the knowledge graph lifecycle

**Tutorial website:** [https://stiinnsbruck.github.io/kgt/](https://stiinnsbruck.github.io/kgt/)

**For industry practitioners:**

An entry point to knowledge graphs. Several pointers for tackling different tasks on knowledge graph lifecycle

**For academics:**

A brief overview of the literature, introduction of some tools, especially in knowledge curation.

**Relevant Literature:**

[https://mindlab.ai/en/publications/](https://mindlab.ai/en/publications/) - An extensive list of the literature on knowledge graphs and their applications with conversational agents
Outline and Agenda

13:30 – 15:00 Part 1
1) Introduction
2) Knowledge Creation
3) Knowledge Hosting

15:00 – 15:30 Coffee Break

15:30 – 17:30 Part 2
4) Knowledge Curation
5) Knowledge Deployment & Discussion
1. What is a Knowledge Graph?

TL;DR:
very large semantic nets that integrate various and heterogeneous information sources to represent knowledge about certain domains of discourse.

Term coined by Google in 2012.
1. What is a Knowledge Graph?

- A graph is a mathematical structure in which some pairs in a set of objects are somehow related. See https://en.wikipedia.org/wiki/Graph_(discrete_mathematics)

- Knowledge: knowledge level vs symbol level
  
  We ascribe knowledge to the actions of an agent.
  
  At the symbol level resides implementations like graph-databases.

- An agent would interpret a knowledge graph to make rational decisions to take actions to reach its goals
1. What is a Knowledge Graph?

But wait, aren’t knowledge bases already doing this?

There are certain characteristic differences between KBs and KGs:

- KBs have a strict separation of TBox and Abox

- KGs do not have a big TBox, but have a very large ABox. There is not much to reason.

- No strict schema: Good for integrating heterogeneous sources, not so much in terms of data quality.
## 1. Knowledge Graphs in the Wild

<table>
<thead>
<tr>
<th>Name</th>
<th>Instances</th>
<th>Facts</th>
<th>Types</th>
<th>Relations</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBpedia (English)</td>
<td>4,806,150</td>
<td>176,043,129</td>
<td>735</td>
<td>2,813</td>
</tr>
<tr>
<td>YAGO</td>
<td>4,595,906</td>
<td>25,946,870</td>
<td>488,469</td>
<td>77</td>
</tr>
<tr>
<td>Freebase</td>
<td>49,947,845</td>
<td>3,041,722,635</td>
<td>26,507</td>
<td>37,781</td>
</tr>
<tr>
<td>Wikidata</td>
<td>15,602,060</td>
<td>65,993,797</td>
<td>23,157</td>
<td>1,673</td>
</tr>
<tr>
<td>NELL</td>
<td>2,006,896</td>
<td>432,845</td>
<td>285</td>
<td>425</td>
</tr>
<tr>
<td>OpenCyc</td>
<td>118,499</td>
<td>2,413,894</td>
<td>45,153</td>
<td>18,526</td>
</tr>
<tr>
<td>Google´s Knowledge Graph</td>
<td>570,000,000</td>
<td>18,000,000,000</td>
<td>1,500</td>
<td>35,000</td>
</tr>
<tr>
<td>Google´s Knowledge Vault</td>
<td>45,000,000</td>
<td>271,000,000</td>
<td>1,100</td>
<td>4,469</td>
</tr>
<tr>
<td>Yahoo! Knowledge Graph</td>
<td>3,443,743</td>
<td>1,391,054,990</td>
<td>250</td>
<td>800</td>
</tr>
</tbody>
</table>

Numerical Overview of some Knowledge Graphs, taken from [Paulheim, 2017]
1. What is a Knowledge Graph?

- Knowledge graphs are not the first attempt for making data useful for automated agents by integrating and enriching data from heterogeneous sources.

- Building knowledge graphs are expensive. Scaling them is challenging.

- A knowledge graph may cost 0.1 - 6 USD per fact [Paulheim, 2018]
1. What is a Knowledge Graph?

Two main entry points for improving the quality of knowledge graphs:

Fixing TBox
- We accept schema.org (and its extensions) as golden standard. No problem here.

Fixing ABox
- This is where knowledge curation comes in.
1. Schema.org

Created, recommended and maintained by four major search engines providers

http://www.schema.org/
1. Schema.org

- Embedded in HTML source
- Microdata
- RDFa
- JSON-LD
1. Schema.org

Hotel
Canonical URL: http://schema.org/Hotel

A hotel is an establishment that provides lodging paid on a short-term basis (Source: Wikipedia, the free encyclopedia, see http://en.wikipedia.org/wiki/Hotel).

See also the dedicated document on the use of schema.org for marking up hotels and other forms of accommodations.

Usage: Between 10,000 and 50,000 domains

<table>
<thead>
<tr>
<th>Property</th>
<th>Expected Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Properties from LodgingBusiness

| amenityFeature   | LocationFeatureSpecification | An amenity feature (e.g. a characteristic or service) of the Accommodation. This generic property does not make a statement about whether the feature is included in an offer for the main accommodation or available at extra costs. |
| audience         | Audience                     | An intended audience, i.e. a group for whom something was created. Supersedes serviceAudience. |
| language         | Text                         | A language someone may use with or at the item, service or place. Please use one of the language |
1. Schema.org

{
    "@context": "http://schema.org",
    "@type": "LocalBusiness",
    "name": "Imbiss-Stand "Wurscht & Durscht"",
    "geo": {
        "@type": "GeoCoordinates",
        "latitude": "47.3006092921797",
        "longitude": "10.9136698539673"
    },
    "address": {
        "@type": "PostalAddress",
        "streetAddress": "Unterer Mooswaldweg 2",
        "addressLocality": "Obsteig",
        "postalCode": "6416",
        "addressCountry": "AT",
        "telephone": "+43 664 / 26 32 319",
        "faxNumber": "",
        "email": "info@wudu-imbiss.at",
        "url": "www.wudu-imbiss.at"
    },
    "description": "Der Imbissstand direkt an der Bundesstraße B 189 in Obsteig verwöhnt die Gäste mit qualitativ hochwertigen "Würschtln" (Wurst) aller Art."
}
# 1. Schema.org

**Event**

An event happening at a certain time and location, such as a concert, lecture, or festival. Ticketing information may be added via the `offers` property. Repeated events may be structured as separate Event objects.

<table>
<thead>
<tr>
<th>Property</th>
<th>Expected Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>about</td>
<td>Thing</td>
<td>The subject matter of the content. Inverse property: <code>subjectOf</code>.</td>
</tr>
<tr>
<td>actor</td>
<td>Person</td>
<td>An actor, e.g. in TV, radio, movie, video games etc., or in an event. Actors can be associated with individual items or with a series, episode, clip. Supersedes <code>actors</code>.</td>
</tr>
<tr>
<td>aggregateRating</td>
<td>AggregateRating</td>
<td>The overall rating, based on a collection of reviews or ratings, of the item. The overall rating, based on a collection of reviews or ratings, of the item.</td>
</tr>
<tr>
<td>attendee</td>
<td>Organization or Person</td>
<td>A person or organization attending the event. Supersedes <code>attendees</code>.</td>
</tr>
</tbody>
</table>

- schema.org is organized as a hierarchy of types and properties
- the data model is derived from RDFS
- `domainIncludes`, `rangeIncludes` instead of `rdfs:domain`, `rdfs:range`
- The ranges are disjunctive
- Types are arranged in multiple inheritance hierarchy
Knowledge Graph Building Process Model
1. Knowledge Graph Building: Task Model

What we will focus on, today
2. Knowledge Creation - Methodology

a.k.a Knowledge Acquisition: “...describes the process of extracting information from different sources, structuring it, and managing established knowledge” - Schreiber et al.

Domain Specification Modeling

- Evaluation and analysis of the annotations
- Annotation development and deployment
- Domain definition and mapping to semantic vocabularies
- Defining a vocabulary based on restricting and extending semantic vocabularies
- Analysis of domain entities and their online representation

Mapping according to domain specifications

Annotation development according to domain specifications

Preparation for modeling

Application of models
2. Knowledge Creation - Methodology

1) **bottom-up**: describes a first annotation process
   a) analysis of a domain’s entities and their (online) representation
   b) defining a vocabulary (potentially by restricting and/or extending an already existing voc.)
   c) “domain definition”, mapping to semantic vocabularies
   d) annotation
   e) evaluation and analysis of annotations
2. Knowledge Creation - Methodology

2) **domain specification modeling**: reflects the results of step 1)
formalize the findings of step 1) in a

- unified
- exchangable
- machine-read and understandable way

⇒ **Domain Specifications**
2. Knowledge Creation - Domain Specifications

“A domain specification is a document, defining syntactic and semantic constraints for schema.org* annotations regarding a specific domain or application” [Holzknecht, 2019]

“[A] domain specification [is] a(n) (extended) subset of properties and restrict[s] the range of those properties to a subset of subclasses of the range defined by schema.org*” [Simsek et al., 2017]

*or any other ontology
(extended: because we not only use schema.org, but also extensions of it if necessary)

Domain Specification are:

- annotation patterns
- a best practice for annotation users
- a “crutch” for annotation laymen
- a means of sharing a common understanding about a domain’s annotation application
2. Knowledge Creation - Domain Specifications

A hotel is an establishment that provides lodging paid on a short-term basis (source: Wikipedia, the free encyclopedia, see http://en.wikipedia.org/wiki/Hotel).

See also the dedicated document on the use of schema.org for marking up hotels and other forms of accommodations.

- DSs are serialized in SHACL
2. Knowledge Creation - Methodology

3) **top-down**: applies models for further knowledge acquisition
   a) mapping according to domain specifications
   b) annotation development according to domain specifications
2. Knowledge Creation - tools - semantify.it

In the “early days” of our KG building efforts: three core questions (by our show-case users*) arised

* our efforts were always driven by educating people (real users, outside of academia, mostly from the industry/tourism) to create their own semantically rich content

1) which vocabulary to use
2) how to create JSON-LD files
3) how to publish those annotations (schema.org in JSON-LD files)

Tool, developed as a research project, grown to a full-stack annotation creation, validation and publication framework!
2. Knowledge Creation - tools - semantify.it

1) Which vocabulary to choose? ⇒ schema.org

Still hundreds of classes and properties in schema.org?

Domain Specifications
- (Extended) subset of schema.org
- Domain expert builds DS files as templates for editor
- Easy to use DS editor
2. Knowledge Creation - tools - semantify.it

2) How to create those JSON-LD files?

- Semantify.it editor & instant annotations
- based on DS
- Inside platform (big DS files)
- or Instant Annotations (IA) portable to every website (based on JS)
- mappers (RocketRML)
- wrapper framework
- semi-automatic

RocketRML ⇒
2. RocketRML - A Quite Scalable RML Mapper [Simsek et al., 2019]

Based on RML [Dimou et al., 2014]:
- Easier to learn RML than a programming language
- Easy sharing
- Mapping can be visualized
- Mapfiles can be faster to write than code
- Easily change mappings
2. RocketRML - A Quite Scalable RML Mapper

- Resolving JOINs is the main bottleneck when it comes to mapping large input files.

- Each TriplesMap is iterated once

- Before starting the mapping process for a TriplesMap, we check whether the TriplesMap is in the join condition of another TriplesMap. If it is, then we get the parent path of the join condition and evaluate it. The value then is cached as path - value pair.
2. RocketRML - A Quite Scalable RML Mapper

- Then we map the data based on the TriplesMap as usual. If there is a join condition encountered during the mapping, then value of the child and path to the parent is cached in the child.

- After everything is mapped, we go through the two caches and join the objects with matching child and parent values.
2. RocketRML - Performance

JSON Format (Size/Time(ms))

XML Format (Size/Time(ms))
2. RocketRML - Source Code

RocketRML - An RML Mapper

View the Project on Github: semantifyit/RocketRML

RocketRML

For the legacy version with the different behavior of the iterator please see this version.

This is a javascript RML-mapper implementation for the RDF mapping language (RML).

Install

`npm install rocketml`

Quick-start

After installation you can to copy `index.js` into your current working directory. Also the `mapfile.ttl` and the input is needed.

`node_index.js`

Starts the execution and the output is then written to `./out.n3`.

Also an example Dockerfile can be seen [here](https://semantifyit.github.io/RocketRML/).

Node.js implementation

Also available as Docker container
2. RocketRML - A Quite Scalable RML Mapper

- Quick demo (https://semantifyit.github.io/rml):

Raw data set (JSON):

```json
1:
  "persons": [
    {
      "firstname": "Elias",
      "lastname": "Kärle",
      "speaks": ["de", "en", "it", "fr", "Tyrolean"]
    },
    {
      "firstname": "Umutcan",
      "lastname": "Simsek",
      "speaks": ["en", "de", "Hessisch"]
    }
  ]
```

Mapping file (YARRRML*):

```yaml
prefixes:
schema: "http://schema.org/
myFunc: "http://myFunc.com/
mappings:
person:
sources: [-'input-jsonpath', '$.persons[*]']
po:
  - [a, schema:Person]
  - [schema:name, $(firstname)]
  - [schema:language, $(speaks.*)]
```

Mapping result:

```json
1:
  "id": "http://example.com/Elias",
  "@type": "Person",
  "language": [
    "de",
    "en",
    "it",
    "fr",
    "Tyrolean"
  ],
  "name": "Elias",
  "@context": {
    "@vocab": "http://schema.org/
  }
},
```

* YARRRML is the yaml-based, human readable, translation of the actual turtle-based RML syntax. (http://rml.io/yarrrml/matey/)
2. Knowledge Creation - tools - semantify.it

2) How to create those JSON-LD files?
   - wrapper framework
2. Knowledge Creation - tools - semantify.it

2) How to create those JSON-LD files?

- semi automatic generation
  - WordPress plugin
  - “guess” the entities of the web page through machine learning
- model trained on entities in our knowledge graph
2. Knowledge Creation - tools - semantify.it

3) How to publish annotations (schema.org in JSON-LD files)?

- copy&paste?
  → pasting content to website is no option for inexperienced users and does not scale

- semantify.it stores all created annotations and provides them over an API
  (http://smtfy.it/sj7Fie2 OR http://smtfy.it/url/http/... OR http://smtfy.it/cid/374fm38dkgi...)

- publication of annotations over JS or into popular CMSs through plugins (Wordpress, TYPO3 etc.)

| GET | /annotation/{annotationId} |
| GET | /annotation/{annotationId}/statistics |
| GET | /organisation/{organisationId}/annotation |
| GET | /website/{websiteId}/annotation |
2. Knowledge Creation - tools - semantify.it

Evaluator:
validation & verification
- **verification** against schema.org
- **verification** against DS
- **validation** against website →
2. Knowledge Creation - tools - semantify.it

Evaluator:
- validation against content of website

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>givenName</td>
<td>Elias Kärle</td>
<td>100</td>
</tr>
<tr>
<td>email</td>
<td><a href="mailto:elias.kaele@st2.at">elias.kaele@st2.at</a></td>
<td>0</td>
</tr>
<tr>
<td>telephone</td>
<td>+4351250753738</td>
<td>0</td>
</tr>
<tr>
<td>image</td>
<td><a href="https://elias.kaele.com/elias.jpg">https://elias.kaele.com/elias.jpg</a></td>
<td>0</td>
</tr>
<tr>
<td>jobTitle</td>
<td>Scientific Assistant</td>
<td>85</td>
</tr>
<tr>
<td>worksFor.department.name</td>
<td>Semantic Technology Institute (STI)</td>
<td>68</td>
</tr>
<tr>
<td>worksFor.name</td>
<td>University of Innsbruck</td>
<td>95</td>
</tr>
<tr>
<td>worksFor.url</td>
<td><a href="https://www.sti-innsbruck.at/">https://www.sti-innsbruck.at/</a></td>
<td>0</td>
</tr>
<tr>
<td>faxNumber</td>
<td>+4351250753738</td>
<td>0</td>
</tr>
</tbody>
</table>
3. Knowledge Hosting

In our context:

“Knowledge is represented in the form of semantically enriched data”

→ **metadata** is added to describe the data by using a (de-facto) **standard vocabulary**, according to the principles of **RDF**
Resource Description Framework (RDF)

1) identify resource with URI: e.g.
http://fritz.phantom.com

2) describe s, p, o

<table>
<thead>
<tr>
<th>Subject</th>
<th>Predicate</th>
<th>Object</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fritz</td>
<td>is a rdf:type</td>
<td>Person schema:Person</td>
</tr>
<tr>
<td>Fritz</td>
<td>has name schema:name</td>
<td>Fritz Phantom schema:Text</td>
</tr>
<tr>
<td>Fritz</td>
<td>lives in xyz:lives</td>
<td>Innsbruck schema:Place</td>
</tr>
<tr>
<td>Fritz</td>
<td>was born in schema:born</td>
<td>1.1.19?? schema:Date</td>
</tr>
<tr>
<td>Fritz</td>
<td>works for xyz:works</td>
<td>Uni Innsbruck schema:Organisation</td>
</tr>
<tr>
<td>Innsbruck</td>
<td>is a rdf:type</td>
<td>town schema:Place</td>
</tr>
<tr>
<td>Innsbruck</td>
<td>is in rdf:type</td>
<td>Tirol schema:Country</td>
</tr>
<tr>
<td>Tirol</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

“Resource”:
Fritz Phantom
Innsbruck
1.1.19??
Uni Innsbruck
what actually are the s, p, o?

Either a URL:
- to identify resources [http://fritz.phantom.com](http://fritz.phantom.com)
- to refer to properties of an ontology [http://schema.org/name/](http://schema.org/name/)
- to refer to types of an ontology [http://schema.org/Person](http://schema.org/Person)

or a literal
- String: “Fritz Phantom”
- Date: “1.1.19??”
- Number: 42
Resource Description Framework

» 2 ways of representation (at least):

1. JSON-LD (for websites)
   
   ```json
   {"@context":"http://schema.org",
   "@type":"Person",
   "@id": "https://fritz.phantom.com",
   "livesIn":"Innsbruck",
   "born":"19??-01-01",
   "worksFor":{"@type":"Organisation",
   "name":"Uni Innsbruck"}}
   ```

2. Graph Database (Knowledge Graph)
3. Knowledge Hosting

Two different approaches for storing semantically annotated data, depending on the use case:

Either as

1) JSON-LD
or as

2) Knowledge Graph
3. Knowledge Hosting

1) Storing as JSON-LD:

**Use-case:** storing semantically annotated data for usage on websites
→ the classical semantify.it use-case
→ many people use semantic annotations exclusively for website for SEO

**Collection/creation:** manual or semi-automatic editing, mapping, wrapper framework (was covered in previous section) or even crawling of annotated web-sites

**Storage:** JSON-based document database, e.g. MongoDB

(JSON-LD is in fact JSON)
3. Knowledge Hosting

1) Storing as **JSON-LD**:

**Pros:**
- seamless and lightning-fast storage and retrieval (through advanced JSON indexing)
- lightweight (little processing power overhead)
- cost effective (starts with powerful free versions)
- good framework integration for web-development
- well documented
- huge community

**Cons:**
- no native RDF reasoning
- reasoning requires extensive programming and processing power overhead
3. Knowledge Hosting

1) Storing as JSON-LD:

Query:
- over an API, through GET request

Summary:
- works very well with tens of millions of JSON-LD files
- we replicate this data periodically into a graph database for “real” Knowledge Graph usage
3. Knowledge Hosting

2) Storing as Knowledge Graph:

Use-case: storing semantically annotated data as a full-fledged Knowledge Graph

→ Open Data repositories in tourism
→ enterprise Knowledge Graphs
→ advanced reasoning needs
→ AI, intelligent assistants

Collection/creation: due to potentially millions of annotation files: mapping, wrapper framework or also crawling of annotated web-sites → semantify.it-broker
3. Knowledge Hosting

**semantify.it-broker:**
- crawling platform to collect annotated data in JSON-LD, Microdata, RDFa
- storage in graph database
- provision of SPARQL UI

**CRAWLING STATISTICS**

- **Crawling took:** 10 minutes
- **Crawling started:** Friday, April 27th 2018, 21:40:16
- **Crawling ended:** Friday, April 27th 2018, 21:50:46
- **Crawled pages:** 3480

**CRAWLING FILTERS**

- Blacklist *sdoType*: BREADCRUMBLIST
- Whitelist markup: JSONLD

**FIND ANNOTATIONS**

- **sdo Types:**
  - BREADCRUMBLIST: 2309
  - PLACE: 23
  - ARTICLE: 20234
  - FOODEVENT: 6
  - MUSICEVENT: 18
  - BUSINESSEVENT: 8
  - EVENT: 4
  - LOCALBUSINESS: 44
  - POSTALADDRESS: 153
  - SPORTSEVENT: 2
  - LOCALBUSINESS: 44
  - LODGINGBUSINESS: 2
  - NEWSARTICLE: 367
  - PERSON: 10
  - TOURISTATTRACTION: 77
  - GEOCOORDINATES: 77
  - LISTITEM: 77

- **Markup:**
  - MICRODATA: 28873
  - JSONLD: 444

- **Total:** 29317

**SAVED ANNOTATIONS**

- **sdo Types:**
  - PLACE: 8
  - FOODEVENT: 6
  - MUSICEVENT: 18
  - BUSINESSEVENT: 8
  - EVENT: 4
  - LOCALBUSINESS: 23
  - LODGINGBUSINESS: 2
  - NEWSARTICLE: 367

- **Markup:**
  - JSONLD: 444

- **Total:** 444
3. Knowledge Hosting

2) Storing as Knowledge Graph:

**Storage:** due to RDF-nature, storage in graph database

with respect to:

- provenance
- historical data
- data duplication

In our current setting:

- historical data is kept in named graphs
- ~13 Billion statements
3. Knowledge Hosting

2) Storing as Knowledge Graph:

Storage: popular triple stores ([https://www.w3.org/wiki/LargeTripleStores](https://www.w3.org/wiki/LargeTripleStores))

<table>
<thead>
<tr>
<th>#</th>
<th>Name</th>
<th># triples tested with</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Oracle Spatial and Graph with Oracle Database 12c</td>
<td>1.08 T</td>
</tr>
<tr>
<td>2</td>
<td>AnzoGraph DB by Cambridge Semantics</td>
<td>1.065 T</td>
</tr>
<tr>
<td>3</td>
<td>AllegroGraph</td>
<td>1+ T</td>
</tr>
<tr>
<td>4</td>
<td>Stardog</td>
<td>50 B</td>
</tr>
<tr>
<td>5</td>
<td>OpenLink Virtuoso v7+</td>
<td>39.8 B</td>
</tr>
<tr>
<td>6</td>
<td>GraphDB™ by Ontotext</td>
<td>17 B</td>
</tr>
</tbody>
</table>
3. Knowledge Hosting

2) Storing as Knowledge Graph:

Pros:
- querying through native SPARQL endpoint

Cons:
- resource intensive
- expensive
3. Knowledge Hosting

2) Storing as **Knowledge Graph:**

**Query:**

- SPARQL

http://graphdb.sti2.at:8080/sparql

**Summary:**

- overhead aside: great for big knowledge graphs
4. Knowledge Curation

- Knowledge Assessment
- Knowledge Cleaning
- Knowledge Enrichment
4. Knowledge Curation - A Simple KR Formalism - TBox

1. Two disjoint and finite sets of type and property names $T$ and $P$.

2. A finite number of type definitions $\text{isA}(t_1,t_2)$ with $t_1$ and $t_2$ are elements of $T$. $\text{isA}$ is reflexive and transitive.

3. A finite number of property definitions:
   - $\text{hasDomain}(p,t)$ with $p$ is an element of $P$ and $t$ an element of $T$.
   - Range definition for a property $p$ with $p$ is an element of $P$, $t_1$ and $t_2$ are Elements of $T$. Simple definition: Global property definition: $\text{hasRange}(p,t_2)$
     - Refined definition: Local property definition: $\text{hasRange}(p,t_2)$ for domain $t_1$, short: $\text{hasLocalRange}(p,t_1,t_2)$
4. Knowledge Curation - A Simple KR Formalism - ABox

1. A countable set of instance identifiers I. i, i1, and i2 are elements of I.

2. Instance assertions: isElementOf(i,t). isElementOf is a special property with build-in semantics. If isA(t1,t2) AND isElementOf(i,t1) THEN isElementOf(i,t2).

3. Property value assertions: p(i1,i2).

4. Equality assertions: isSameAs(i1,i2). We allow another build-in property to express identity of instances. It is symmetric, reflexive, and transitive.
4. Knowledge Curation - Knowledge Assessment

- First step to improve the quality of a KG: Assess the situation

- Closely related to data quality literature

- Various dimensions for data quality assessment introduced [Batini & Scannapieco, 2006], [Färber et al., 2018], [Pipino et al., 2002], [Wang, 1998], [Wang & Strong, 1996], [Wang et al., 2001], [Zaveri et al., 2016]
4. Knowledge Curation - Knowledge Assessment

1. accessibility
2. accuracy (veracity)
3. appropriate amount
4. believability
5. completeness
6. concise representation
7. consistent representation
8. cost-effectiveness
9. easy of manipulating
10. easy of operation
11. easy of understanding
12. flexibility
13. free-of-error
14. interoperability
15. objectivity
16. relevancy
17. reputation,
18. security,
19. timeliness (velocity),
20. traceability,
21. understandability,
22. value-added, and
23. variety

**fitness for use**
4. Knowledge Curation - Knowledge Assessment Tasks

- Two core assessment dimensions for Knowledge Graphs
  - Correctness
  - Completeness

- Three quality issue sources:
  - Instance assertions
  - Property value assertions
  - Equality assertions
4. Knowledge Curation - Knowledge Assessment Tools

- WIQA (Web Information Quality Assessment Framework) [http://wifo5-03.informatik.uni-mannheim.de/bizer/wiqa/ [Bizer and Cyganiak, 2009]:

  Allows defining policies to filter triples in a graph

- SWIQA (Semantic Web Information Quality Assessment Framework) [Fürber & Hepp, 2011]:

  A set of SPARQL-based rules to assess data quality
4. Knowledge Curation - Knowledge Assessment Tools

- LINK-QA [Guéret et al., 2012]

  Benefits from network features to assess data quality (e.g. counting open chains to find wrongly asserted isSameAs relationships)

- Sieve [Mendes et al., 2012] https://github.com/wbsg/ldif/

  Uses data quality indicators, scoring functions and assessment metrics
4. Knowledge Curation - Knowledge Assessment Tools

- Validata [Hansen et al., 2015]  
  https://github.com/HW-SWeL/Validata

  An online tool check the conformance of RDF graphs against ShEx (Shape Expressions)

- Luzzu (A Quality Assessment Framework for Linked Open Datasets) [Debattista et al., 2016]  
  https://eis-bonn.github.io/Luzzu/downloads.html

  Allows declarative definitions of quality metrics and produces machine-readable assessment reports based on Dataset Quality Vocabulary
4. Knowledge Curation - Knowledge Assessment Tools

- RDFUnit [Kontokostas et al., 2014]  [https://github.com/AKSW/RDFUnit/](https://github.com/AKSW/RDFUnit/):  
  A framework that assesses linked data quality based on test cases defined in various ways (e.g. RDFS/OWL axioms can be converted into constraints)

- SDType [Paulheim & Bizer, 2013]  [https://github.com/HeikoPaulheim/sd-type-validate](https://github.com/HeikoPaulheim/sd-type-validate)
  Uses statistical distributions to predict the types of instances. Incoming and outgoing properties are used as indicators for the types of resources.
4. Knowledge Curation - Knowledge Assessment Tools - Example

● **Sieve for Data Quality Assessment**
  ○ **Data Quality Indicators**: Various type of (meta)data that can be used to assess data quality e.g. data about the dataset provider, user ratings
  ○ **Scoring Functions**: A set of functions that help the calculation of assessment metrics based on the indicators
  ○ **Assessment Metrics**: Metrics like relevancy, timeliness that help users to assess the quality for an intended use
  ○ **Aggregate Metrics**: Allow users to aggregate new metrics based on simple assessment metrics.
### 4. Knowledge Curation - Knowledge Assessment Tools - Example

<table>
<thead>
<tr>
<th>SCORING FUNCTION</th>
<th>EXAMPLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>TimeCloseness</td>
<td>measures the distance from the input date to the current (system) date. Dates outside the range receive value 0, and dates that are more recent receive values closer to 1.</td>
</tr>
<tr>
<td>Preference</td>
<td>assigns decreasing, uniformly distributed, real values to each graph URI provided as a space-separated list.</td>
</tr>
<tr>
<td>SetMembership</td>
<td>assigns 1 if the value of the indicator provided as input belongs to the set informed as parameter, 0 otherwise.</td>
</tr>
<tr>
<td>Threshold</td>
<td>assigns 1 if the value of the indicator provided as input is higher than a threshold informed as parameter, 0 otherwise.</td>
</tr>
<tr>
<td>IntervalMembership</td>
<td>Assigns 1 if the value of the indicator provided as input is within the interval informed as parameter, 0 otherwise.</td>
</tr>
</tbody>
</table>

Assessment Metrics in Sieve
4. Knowledge Curation - Knowledge Cleaning

- The actions taken to improve the correctness of a knowledge graph.

- Two major steps:
  - Error detection
  - Error correction
4. Knowledge Curation - Knowledge Cleaning Tasks

Detection and correction of wrong instance assertions: isElementOf(i.t)

<table>
<thead>
<tr>
<th>Error</th>
<th>Correction</th>
</tr>
</thead>
<tbody>
<tr>
<td>i is not a proper instance identifier</td>
<td>Delete assertion or correct i</td>
</tr>
<tr>
<td>i1 is not a valid instance identifier</td>
<td>Delete assertion or correct t.</td>
</tr>
<tr>
<td>Instance assertion is semantically incorrect</td>
<td>Delete assertion or find proper t.</td>
</tr>
</tbody>
</table>
4. Knowledge Curation - Knowledge Cleaning Tasks

Detection and correction of wrong property value assertions \( p(i_1, i_2) \)

<table>
<thead>
<tr>
<th>Error</th>
<th>Correction</th>
</tr>
</thead>
<tbody>
<tr>
<td>( p ) is not a valid property</td>
<td>Delete assertion or correct ( p )</td>
</tr>
<tr>
<td>( i_1 ) is not a valid instance identifier</td>
<td>Delete assertion or correct ( i_1 )</td>
</tr>
<tr>
<td>( i_1 ) is not in any domain of ( p )</td>
<td>Delete assertion or add assertion ( \text{isElementOf}(i_1, t) ) where ( t ) is in a domain of ( p )</td>
</tr>
</tbody>
</table>

Karlsruhe I Kärle & Simsek I September 9, 2019
4. Knowledge Curation - Knowledge Cleaning Tasks

Detection and correction of wrong property value assertions $p(i_1,i_2)$

<table>
<thead>
<tr>
<th>Error</th>
<th>Correction</th>
</tr>
</thead>
<tbody>
<tr>
<td>$i_2$ is not a valid instance identifier</td>
<td>delete assertion or correct $i_2$</td>
</tr>
<tr>
<td>$i_2$ is not in any range of $p$ where $i_1$ is an element of a domain of $p$.</td>
<td>Delete assertion or Add assertion isElementOf($i_1,t_1$) given that hasLocalRange($t_1,p,t_2$) and isElementOf($i_2,t_2$) or Add assertion isElementOf($i_2,t_2$) given that hasLocalRange($t_1,p,t_2$) and isElementOf($i_1,t_1$)</td>
</tr>
<tr>
<td>Property assertion is semantically incorrect.</td>
<td>Delete assertion or define a proper $i_2$ or find a better $p$ or better $i_1$</td>
</tr>
</tbody>
</table>
4. Knowledge Curation - Knowledge Cleaning Tasks

Detection and correction of wrong equality assertions isSameAs(i1,i2)

<table>
<thead>
<tr>
<th>Error</th>
<th>Correction</th>
</tr>
</thead>
<tbody>
<tr>
<td>i1 is not a valid instance identifier</td>
<td>Delete assertion or correct i1</td>
</tr>
<tr>
<td>i2 is not a valid instance identifier</td>
<td>Delete assertion or correct i2</td>
</tr>
<tr>
<td>Equality assertion is semantically wrong</td>
<td>Delete assertion or loosen the semantics (e.g. replace by a skos operator)</td>
</tr>
</tbody>
</table>
4. Knowledge Curation - Knowledge Cleaning Tools

- **HoloClean** [Rekatsinas et al., 2017] [https://hazyresearch.github.io/snorkel/blog/holoclean.html](https://hazyresearch.github.io/snorkel/blog/holoclean.html)

  An error detection and correction tool based on integrity constraints to identify conflicting and invalid values, external information to support the constraints, and quantitative statistics to detect outliers.

- **KATARA** [Chu et al., 2015]

  Learns the relationships between data columns and validate the learn patterns with the help of existing Knowledge Bases and crowd, in order to detect errors in the data. Afterwards it also suggests possible repairs.
4. Knowledge Curation - Knowledge Cleaning Tools

  
  Uses statistical distribution to detect erroneous statements that connect two resources. The statements with less frequent predicate-object pairs are selected as candidates for being wrong.

- **SHACL** [https://www.w3.org/TR/shacl/](https://www.w3.org/TR/shacl/) and ShEx [https://shex.io/shex-semantics/index.html](https://shex.io/shex-semantics/index.html)
  
  Two approaches that aim to verify RDF graphs against a specification (so called shapes). For a comparison of two approaches, see Chapter 7 in [Gayo et al., 2017]
4. Knowledge Curation - Knowledge Cleaning Tools

- **LOD Laundromat** [Beek et al., 2014] [http://lodlaundromat.org/](http://lodlaundromat.org/)
  
  Detects and corrects syntactic errors (e.g. bad encoding, broken IRIs), replaces blank nodes with IRIs, removes duplicates in dirty linked open data and re-publishes it in a canonical format.

- **TISCO** [Rula et al., 2019]
  
  A framework that tries to identify the time interval where a statement was correct. It uses external knowledge bases and the web content to extract evidence to assess the validity of a statement for a time interval.
4. Knowledge Curation - Knowledge Enrichment

Improve the completeness of a knowledge graph by adding new statements

- Consists of following steps
  - Identifying new knowledge sources
  - Integration of TBox
  - Integration of Abox
4. Knowledge Curation - Knowledge Enrichment

- Identifying knowledge sources
  - Open sources (e.g. LOD) - may be automated to some extent
  - Proprietary sources - usually very hard automate

- Integration of TBox
  - We assume that all data sources are mapped to schema.org
  - Non-RDF sources can be also mapped with the techniques described in Knowledge Creation
4. Knowledge Curation - Knowledge Enrichment

- Integration of ABox
  - Issue-1: Identifying and resolving duplicates
  - Issue-2: Invalid property assertions (e.g. multiple disjoint values for unique properties, domain and range violations)
Different names for the same problem! [Getoor et al., 2012]

Tackling issues:

- Entity resolution: Derive new isSameAs(i1,i2) assertions and aligning their property assertions
- Conflict resolution: Resolve conflicting property assertions
- Enrichment also has implications towards cleaning!
4. Knowledge Curation - Knowledge Enrichment Tasks

- Identifying and resolving duplicates
- Resolving conflicting property assertions

can be realized by

- addition of missing instance assertions: isElementOf(i,t)
- addition or deletion of property value assertions: p(i1,i2)
- addition of missing equality assertions: isSameAs(i1,i2)
4. Knowledge Curation - Knowledge Enrichment Tools

Duplication detection and resolution tools

- Dedupe: [https://github.com/dedupeio/dedupe](https://github.com/dedupeio/dedupe)

  A python library that uses machine learning to find duplicates in a dataset and to link two datasets.


  Uses various similarity metrics to detect duplicates in a dataset or link records between two datasets based on a given configuration. The configuration parameters can be
4. Knowledge Curation - Knowledge Enrichment Tools

Duplication detection and resolution tools

- Legato [Achichi et al., 2017] [https://github.com/DOREMUS-ANR/legato](https://github.com/DOREMUS-ANR/legato)

  A recording linkage tool that utilizes *Concise Bounded Description* of resources for comparison. *https://www.w3.org/Submission/2004/SUBM-CBD-20040930/#r6*

- LIMES [Ngomo & Auer, 2011] [https://github.com/dice-group/LIMES](https://github.com/dice-group/LIMES)

  A link discovery approach that benefits from the metric spaces (in particular triangle inequality) to reduce the amount of comparisons between source and target dataset.
4. Knowledge Curation - Knowledge Enrichment Tools

Duplication detection and resolution tools

- SERIMI [Araújo et al., 2011] [https://github.com/samuraraujo/SERIMI-RDF-Interlinking](https://github.com/samuraraujo/SERIMI-RDF-Interlinking)
  
  A link discovery tool that utilizes string similarity functions on “label properties” without a prior knowledge of data or schema

- SILK [Volz et al., 2009] [http://silkframework.org/](http://silkframework.org/)
  
  A link discovery tool with declarative linkage rules applying different similarity metrics (e.g. string, taxonomic, set) that also supports policies for the notification of datasets when one of them publishes new links to others.
4. Knowledge Curation - Knowledge Enrichment Tools

Conflict resolution tools

- FAGI [Giannopoulos et al., 2014]  [https://github.com/GeoKnow/FAGI-gis](https://github.com/GeoKnow/FAGI-gis)
  A framework for fusing geospatial data. It suggests fusion strategies based on two datasets with geospatial data and a set of linked entities.

- KnoFuss [Nikolov et al., 2008]  [http://technologies.kmi.open.ac.uk/knofuss/](http://technologies.kmi.open.ac.uk/knofuss/)
  A framework that allows the application of different methods on different attributes in the same dataset for identification of duplicates and resolves inconsistencies caused by the fusion of linked instances.
4. Knowledge Curation - Knowledge Enrichment Tools

Conflict resolution tools

- **ODCleanStore [Knap et al., 2012]**
  
  A framework that contains a fusion module that allows users to configure conflict resolution policies based on different functions (e.g. AVG, MAX, CONCAT) that can be applied on conflicting property values.

- **Sieve [Mendes et al., 2012]**
  
  Sieve has a data fusion module that supports different fusion functions on selected property values. It also utilizes the assessment values from the assessment module in the fusion process.
4. Knowledge Curation - Knowledge Enrichment Tools - Demo

Duplication detection and resolution with Duke
5. Knowledge Deployment

- training of ML models based on KGs
  - due to the RDF nature data in KGs is semantically described
  - good training data for ML models
- conversational agents
  - chatbots
  - intelligent personal assistants
  - question answering over LinkedData
- OpenData sharing platforms
  - currently Open(Government)Data often makes little sense (scanned pdfs, weird spreadsheets, csv, ...)
  - LinkedData is self explaining (see lod-cloud https://lod-cloud.net)
5. Knowledge Deployment - discussion

- are you using KGs in your enterprise / research already?

- are you planning to?

- where do you see the potential

- where do you see challenges / risks?
References


References


References


References


References


