3. Vagueness in Language

Outline

1. An Introductory Example
2. Stop Word Elimination
3. Stem and/or Base Form Reduction
4. Compound Word Identification
5. Terminological Control
6. A Language-independent Approach
7. Marking by means of Metadata

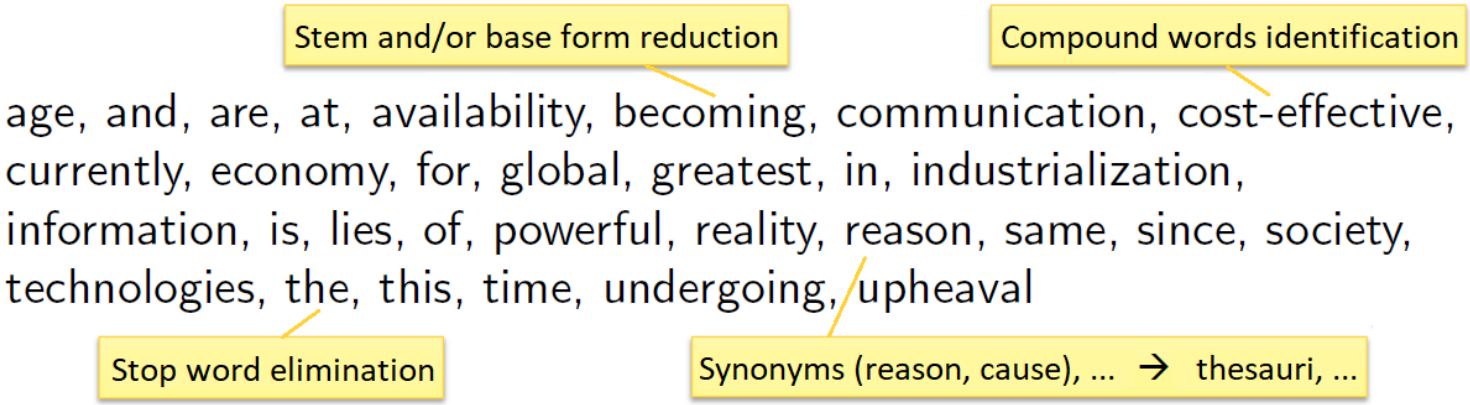
$\Rightarrow$ Depending on the language $\Rightarrow$ different effort!

Terminology

- Word
  - strong reference to the ‘string’
  - thus rather refers to the syntactic level
- Term
  - addresses more strongly the meaning
- Concept
  - expresses stronger reference to semantics
  - usually corresponds to a group of words that are highly related in content and have a content concept behind

3.1 An Introductory Example

- Text:
  - The economy and society are currently undergoing the greatest upheaval since industrialization. The reason for this lies in the global availability of powerful and at the same time cost-effective information and communication technologies. The information age is becoming reality.
- Possible representation (set of words):



3.2 Stop Word Elimination

- Goal:
  - ignoring terms that do not contribute to the ‘semantics’ of the documents
- Effects:
  - reduction of storage space consumption
  - thus also improving the performance of the matching algorithms
  - improvement of recall and precision (e.g. in vector space model)
- Approaches to implementation:
  - managing a stop word list
  - elimination of high and low frequency terms

Stop word list English example (571 words):

a a's able about above according accordingly across actually after afterwards again against ain't all allow allows almost alone along already also although always am among amongst an and another any anybody anyhow anyone anything anyway anyways anywhere apart appear appreciate appropriate are aren't around as aside ask asking associated at available away awfully b be became because become becomes becoming been before beforehand behind being believe below beside besides best better between beyond both brief but by c c'mon c's came can can't cannot cant cause causes certain certainly changes clearly co com come comes concerning consequently consider considering contain containing contains corresponding could couldn't course currently d definitely described despite did didn't different do does doesn't doing don't done down downwards during ...

`http://members.unine.ch/jacques.savoy/clef/englishST.txt`

Problems with stop word lists

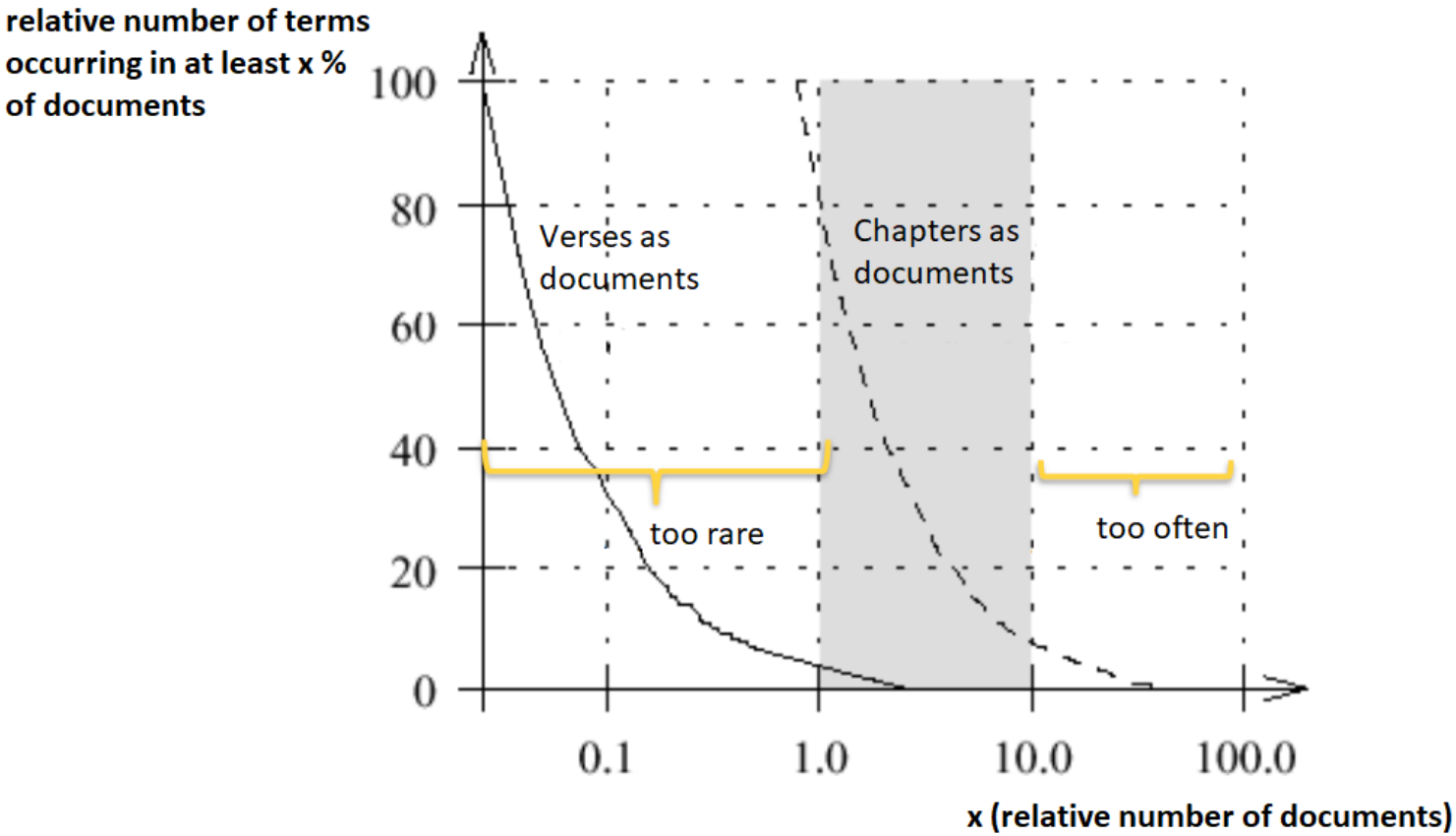
- Creation and maintenance effort:
  - but: there are comprehensive stop word lists for all common languages
- If necessary, adaptation to the domain:
  - term ‘computer’ could be a useful stop word for computer science journals

Elimination of high and low frequency terms

- In a publication, Crouch suggests [Cro90]:
  - not to consider terms that occur in less than 1% of all documents because they are too specific, and
  - not to consider terms that occur in more than 10% of all documents because they are too general
- Requires existence of a representative document collection

Effect of document size

Example: chapters and verses of the New Testament of the Bible



3.3 Stem and/or Base Form Reduction

- Inflection forms
  - conjugation: go - went - gone
  - declension: house - houses
- Same stem in different derivational forms
  - "connect" is stem of "connected", "connecting", "connection"

Matching should refer to word stems

$\Rightarrow$ better recall

Use within the scope of query processing

- Extension of the query, or
- General mapping of terms to the base or stem form

Query extension

- All word forms are preserved in the query
- Query is extended by all conceivable word forms, e.g. by thesaurus or lexicon

- **Advantage:** information content of the query remains fully intact
- **Disadvantage:** query can become extensive
  $\Rightarrow$ performance losses

Better is general mapping of terms to base and stem form

Mapping to the base and stem form

Reduction to the base or stem form when converting queries/documents to the respective representation

- **Advantages:**
  - reduced number of terms to be managed
  - $\Rightarrow$ increased system performance
  - tracing to base and stem form is easier than query expansion

Disadvantage:
specific search for word form becomes impossible $\Rightarrow$ loss of precision

Reduction to the base form

- Words ⇒ basic grammatical form
  - nouns ⇒ nominative (subject) singular
  - verbs ⇒ infinitive
- Procedure: removal of inflection endings and mapping to existing words

Example: applies ⇒ appl ⇒ apply

Stem form reduction

- Traces words back to their stem
- Stem does not have to be a word
- Stem for verb and corresponding nouns is the same

Example: computer, compute, computation, computerization $\Rightarrow$ comput

Terms according to Kuhlen

Kuhlen [Kuh77] distinguishes three procedures for base and stem form reduction:

- Basic lexicographic form
- Basic formal form
- Stem form

Basic lexicographic form

- Form in which the word can be found in a dictionary
- Reversal of graphemic changes in the base form that may have occurred due to inflection

Basic formal form

Basic formal form according to Kuhlen:
"... word fragments in which the "normal" English and foreign language (mainly Latin) inflectional endings are removed without recoding the resulting word fragments."

Stem form

Stem form according to Kuhlen:

- Strings created by *deflection* and removal of derivative endings
- Strings are to be unified by *recoding* as far as possible

Formal, lexical base form and stem form by example

Basic formal form	Text words	Basic lexical form	Stem form
absorb **3**	absorb absorbed absorbing absorbs	absorb	absorb ... ...
	absorber **1** absorbers	absorber **2** ...	... **4** ...
absorbab ...	absorbable absorbably	absorbable ...	... ...
absorbanc ...	absorbance absorbances	absorbance ...	... ...
... ...	absorbancy absorbancies	absorbancy ...	... ...
absorbant	absorbant absorbants absorbantly	absorbant	... ...
absorbtion ... absorbtiv ...	absorbtion absorbtions absorbtively absorbtive	absorbtion ... absorbtive ...	... ...

Methods for base and stem form reduction

- Truncation
- Rules
- Dictionaries

Usage depends on language

- English: weak inflection $\Rightarrow$ rules
- Italian: more inflected $\Rightarrow$ many rules
- German: dictionaries

Methods based on simple truncation

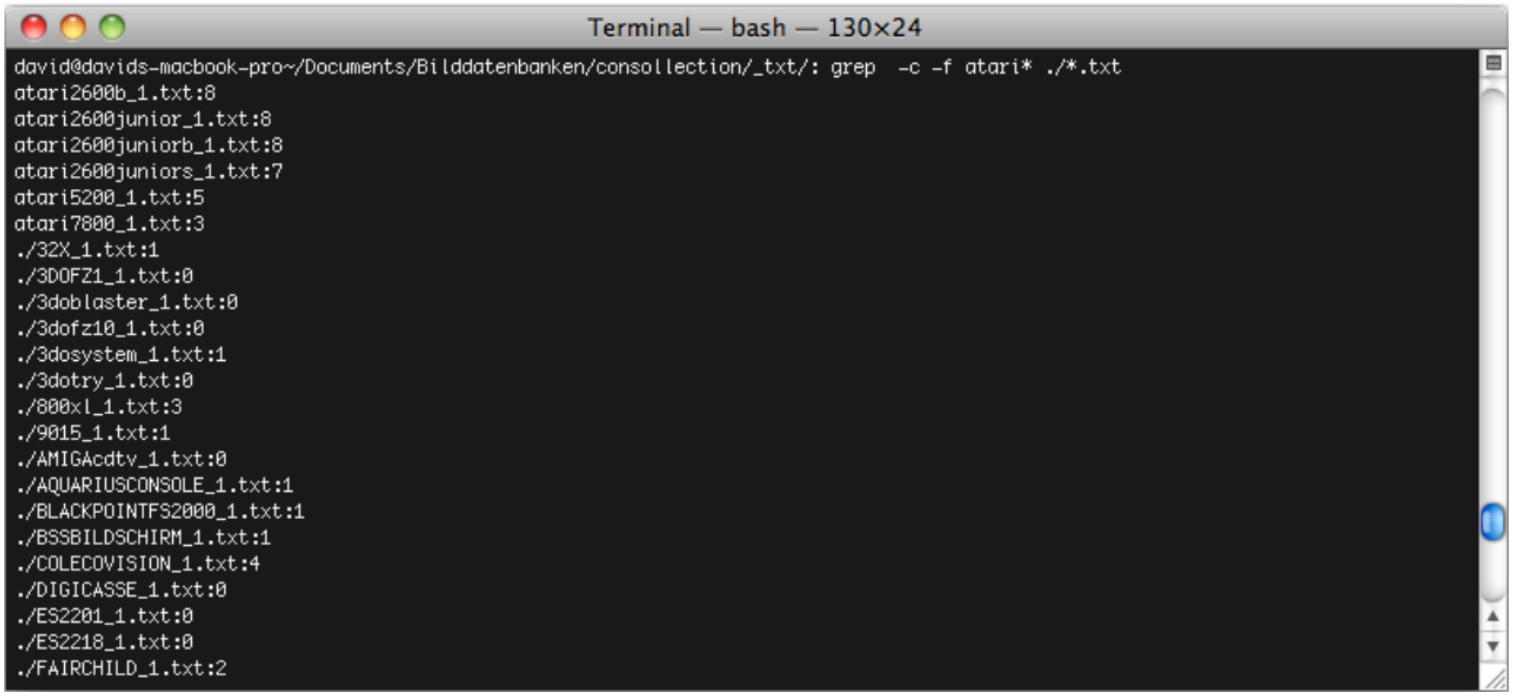
- Truncation explicitly in the query
- For example in form of regular expressions
  - regular expressions in UNIX (according to [Gul88]), for example with *grep*:

Meaning	**Expression**
Any single character	.(*dot*)
Any string (even the empty one)	.*
Any repetition of the preceding character (even none)	*
Any repetition of the preceding character (at least 1)	+
0 or 1 repetition of the preceding character	?
One of the characters from ...	[...]
One of the characters from the range ...	$[a-e]$
One of the characters from the ranges ...	$[a-eh-x]$
All characters except ...	[^ . . .]
Escape symbol	\

Methods based on simple truncation

- Simple query with truncation:
  compute.* AND m[oui]{1,2}[sc]e
- **Disadvantages:**
  - complex query formulation
  - syntactic rules of natural language are not supported

Un*x-Tool *grep*



Lovins algorithm for basic form reduction

- Rule-based algorithm
- Suitable for texts in English
- Works in two steps
  - removal of the endings
  - transformation of the remaining endings
- Not best ending of the word is found
- But: consistency, i.e. same treatment of all words

Procedure of the Lovins algorithm

- Removal of endings using the following table
- Cases
  - A: no condition
  - B: remaining stem has at least 3 characters
  - C: remaining stem has at least 4 characters

Removal of word endings

Length	Ending	Condition
11	alistically	B
	arizability	A
	izationally	B
10	antialness	A
	arisations	A
	arizations	A
	entialness	A
9	allically	C
	antaneous	A
4	able	A
	ably	A
	ages	B
	ally	B
3	ism	B
1	e	A

Procedure of the Lovins algorithm

Further steps:

- If word ends with *"s"* and no *"s"* before it
  $\Rightarrow$ Remove the *"s"*
  - *stems* becomes *stem*
  - but *stress* remains *stress*

Procedure of the Lovins algorithm

Further steps:

- If word ends with *"es"*
  $\Rightarrow$ Remove the final *"s"*
  - *places* becomes *place*
  - *likes* becomes *like*
  - *theses* becomes *these* (Error!)
  - *indices* becomes *indice* (Error!)
  - *syntheses* becomes *synthese* (Error!)

Problem: Words of Greek origin with singular form *"-is"* (plural form *"-es"*)

Procedure of the Lovins algorithm

Further steps:

- Convert *"-iev"* to *"-ief"* and *"-metr"* to *"-meter"*
  - *believable* becomes *believ* and then *belief*

Procedure of the Lovins algorithm

Further steps:

- If word ends with *"ing"* and is preceded by more than one letter and no *"th"*
  $\Rightarrow$ Delete *"ing"*
  - *thinking* becomes *think*
  - *singing* becomes *sing*
  - *sing* becomes *sing* (no change)
  - *thing* becomes *thing* (no change)
  - *preceding* becomes *preced* (Error!, no word)
- Troubleshooting: If word after reduction ends with *"et"*, *"ed"*, *"es"*
  $\Rightarrow$ Append *"e"*

Procedure of the Lovins algorithm

Further steps:

- If word ends with *"ed"*, preceded by a consonant and preceded by at least 1 character
  $\Rightarrow$ Delete *"ed"*
  - *ended* becomes *end*
  - *red* becomes *red*
  - *proceed* becomes *proceed*
  - *proceeded* becomes *proceed*

Procedure of the Lovins algorithm

Further steps:

- If a word ends with *bb*, *dd*, ..., *tt* after removing the ending
  $\Rightarrow$ Remove one of the duplicate characters
  - *embedded* becomes *embedd* and then *embed*

Procedure of the Lovins algorithm

Further steps:

- If word with more than 5 characters ends with *"ion"*
  $\Rightarrow$ Remove *"ion"*
- If last character of stem consonant and preceding one vowel
  $\Rightarrow$ Add *"e"*
  - *direction* becomes *direct*
  - *polution* becomes *polute*
  - *plantation* becomes *plantate* (Error!)
  - *zion* becomes *zion*
  - *scion* becomes *scion*
  - *cation* becomes *cate* (Error!, because cation is artificial word)

Lovins algorithm

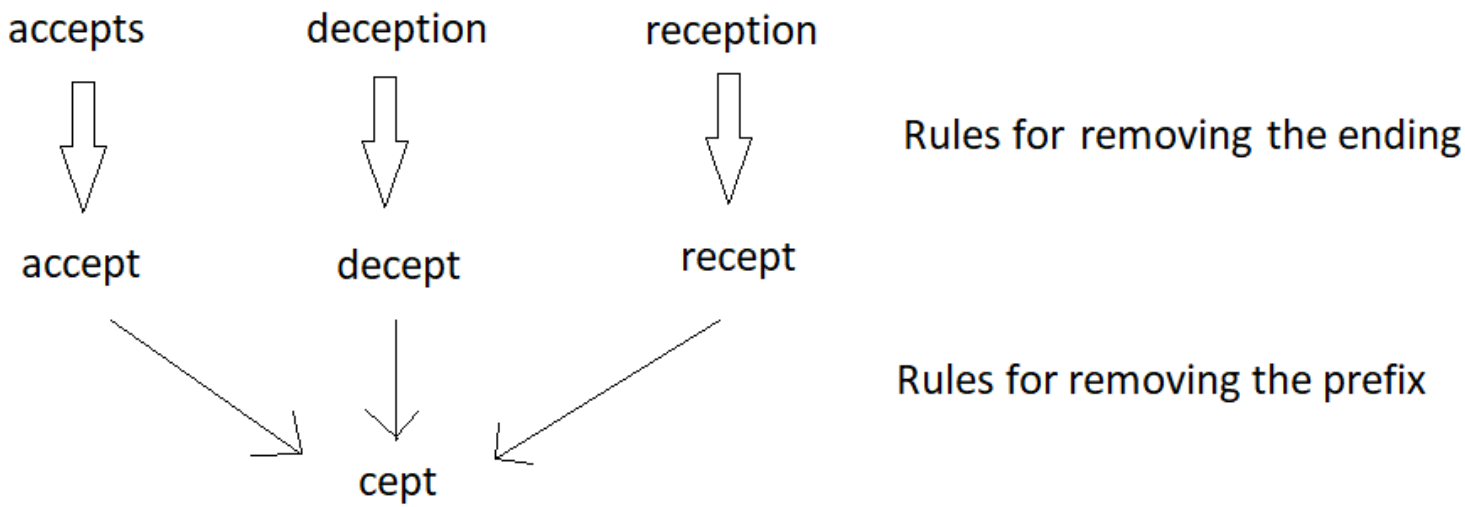
Conclusion:

- English language needs about 10 to 20 rules
- Exceptions for irregular verbs necessary
- Rules should be applicable iteratively
  - *directions* becomes *direction* becomes *direct*

Consideration of prefixes

Prefixes should not be removed

- Example: although same Latin stem, prefixes express different semantics



Other rule-based algorithms

- Porter's algorithm [Por80]
  - used very often
  - same idea as Lovins algorithm
  - but more complex rules
  - e.g. in text search of PostgreSQL 8.4
- Further stemming algorithms available for English, such as Lancaster Stemming Algorithm
- Martin Porter's project with ready algorithms and framework for self-development
  - `http://snowball.tartarus.org/`

Methods based on dictionaries

- Rule-based methods only for weakly inflected languages like English
- For strongly inflected languages (German language)
  $\Rightarrow$ dictionary
  - inflection form $\rightarrow$ base form
    - *lief* $\rightarrow$ *laufen*
    - *Häuser* $\rightarrow$ *Haus*
  - derivative form $\rightarrow$ base form
    - *Lieblosigkeit* $\rightarrow$ *lieblos*
    - *Berechnung* $\rightarrow$ *rechnen*

Problem: maintenance of the dictionary, technical vocabulary often necessary

3.4 Compound Word Identification

- Search for “Bundeskanzlerwahl”
- Document, however, contains “die Wahl des Bundeskanzlers”
- What about document “... die Wahl des Bundeskanzlers fiel auf Saumagen mit Kraut ...”?

Approaches to compound words identification in German

- Ignoring the problem, i.e. only reduction to basic and stem form
  - terms are for example “Wahl”, “Bundeskanzler” and “Bundeskanzlerwahl”
  - problem: search for “Wahl” in document “Bundeskanzlerwahl” not successful
    $\Rightarrow$ Recall suffers
- Decomposing compound words into components
  - recall increases
  - search for “Bundeskanzlerwahl” factually impossible

Approaches to compound words identification in English

- No compound words available
  $\Rightarrow$ Words are already disassembled
  - *compound terms* and *phrases* carry their own meaning
  - e.g. "Information Retrieval" or "Artificial Intelligence"
    $\Rightarrow$ Precision suffers

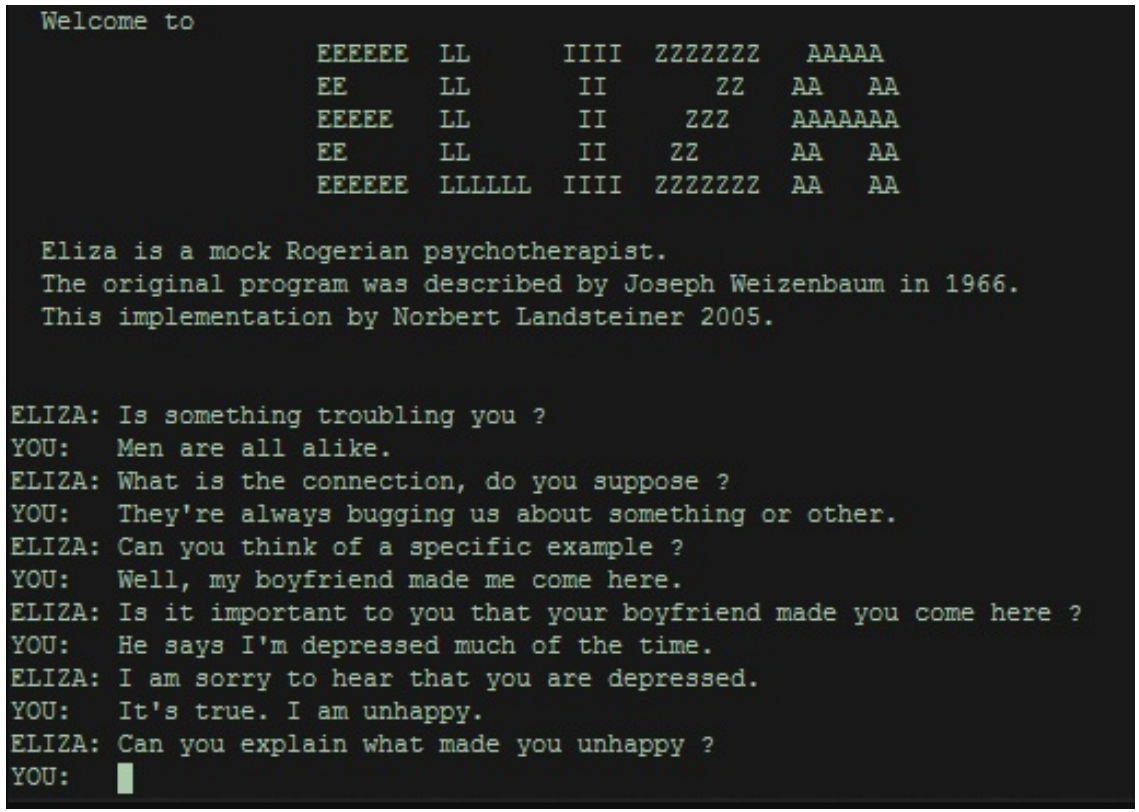
Identifying compound words in English

- 1. Approach
  - Compound word: words A and B with a maximum of n words in between
  - Refinement: consideration of order and sentence boundaries
  - Note: “college junior”, “junior college” and “junior in college” express completely different meanings

Identifying compound words in English

- 2. Approach (Natural Language Processing)
  - Parser determines grammatical sentence structure
  - Problems:
    - not all sentences conform to the grammar
    - complex sentence forms
    - slow algorithms
  - Nevertheless, usable parsers exist, see for example in [Str94]

Weizenbaums ELIZA (1966)



Compound words - Identification by the Darmstadt indexing approach

Simple, robust method for identifying compound words [Lus86]

- Only for German (examples in English)
- Starting point is compound word dictionary
  - obtained automatically by analysis of numerous texts
  - long words were identified and then decomposed
- Analysis of the text
  - occurrence of a component of a compound word
    $\Rightarrow$ test if remaining component resides within maximum distance

Compound words - Identification by the Darmstadt indexing approach

Criteria for describing a compound word:

1. Spacing between components (all words, stop words only, or relevant words only)
2. Order of components
3. Components in the same sentence
4. For each component: equality with respect to base form or stem form

$\Rightarrow$ Probability of syntactically correct identification of compound word can be calculated

Experimental studies have shown that only the first and fourth of the above characteristics significantly influence identification reliability

Compound words - Identification by the Darmstadt indexing approach

Identified components in the sample text

- TUNNEL JUNCTION
- TUNNEL CURRENT
- ELECTRICAL CONDUCTIVITY
- ALUMINIUM SUBSTRATES
- MEASURE ELECTRICAL
- FILM COEFFICIENT

The first three are correct, the rest wrong

Current-voltage spectra of metal/oxide/SnTe diodes. Pt. 1

In metal/oxide/SnTe tunnel junctions (where the oxide is Al_2O_3 or SiO_2 and the metal is lead or aluminium) on BaF_2 or $NaCl$ substrates the tunnel current $I(U)$ and its derivatives $I'(U)$ and $I''(U)$ were measured at 4.2 K. Additionally the Hall coefficient and electrical conductivity of the monocrystalline SnTe films were determined at the same temperature. The pronounced oscillations in I'' suggest the existence of a quantum size effect in the very thin SnTe films in several cases, although this is complicated by various other processes. The most important features of the different types are discussed briefly.

Terms and concepts

- **Precombination:**
  index language contains compound words, such as “cervical vertebra fracture” or “European single market”
- **Precoordination:**
  word linking while indexing for creating the representations, such as “cervical vertebra” + “fracture” $\Rightarrow$ “cervical vertebra fracture”
- **Postcoordination:**
  word link formed during the search using Boolean or other operators, such as “Europe AND single market”

3.5 Terminological Control

- Comes from library systems
- Goal: avoids ambiguity by using "unambiguous" terms
- Manual indexing: two-step process
  1. recognizing the essence of a text to be retrieved
  2. reproduce these essences in an adequate form
- Usage of an artificial language with unique preferred naming

Terminological control

Indexing language:

- Artificial language
- Serves to represent the essence of a text
- Descriptors: elements of the indexing language

- Terminology control: rules and activities that unambiguously define a descriptor and eliminate ambiguities and vagueness

Synonym control

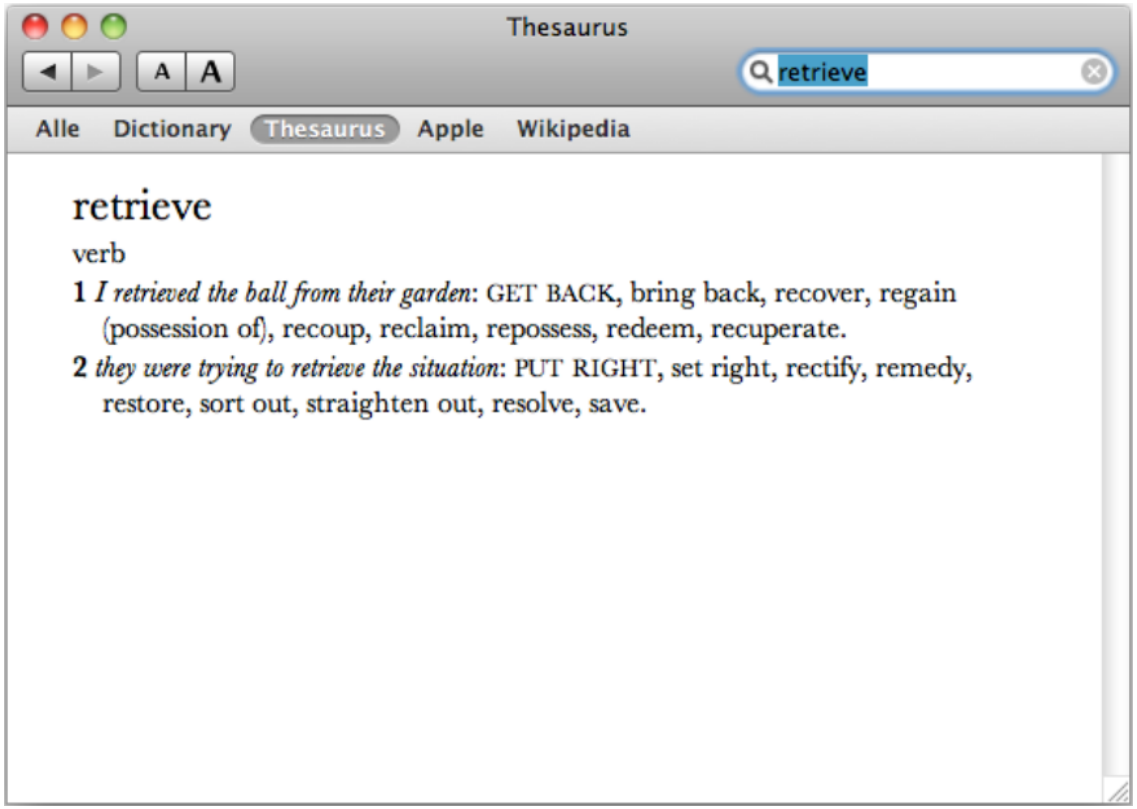
- Grouping of terms into equivalence classes
- Types of synonyms:
  - spelling variants: ‘fiber’ and ‘fibre”
  - different connotations, linguistic styles, areas of distribution: ‘nearsightedness’ and ‘myopia’, ‘gasoline’ and ‘petrol’ or ‘salt’ and ‘sodium chloride’
  - quasi synonyms: ‘mist’ and ‘fog’ or ‘dive’ and ‘plunge’
- Descriptor (preferred term): ‘steed’, ‘mare’ and ‘nag’ $\Rightarrow$ ‘horse’

Synonym control

Use of thesauri

- Equivalence classes of synonyms and
- Terms with little or irrelevant differences in meaning
  - different specificity: ‘philology’ and ‘linguistics’
  - antonyms: ‘hard’ and ‘soft’
  - special subterm: ‘winter wheat’ and ‘wheat’
  - equating verb and noun: ‘participate’ and ‘participation’ (stemming)

General Thesaurus



Polysem control

Allocation of ambiguous terms to several equivalence classes

- Homographs: differentiation according to *linguistic emphasis*
  "sow" [səʊ] - to plant seeds
  "sow" [saʊ] - a female pig
- Polysemy: differentiation according to the *context*
  "bank" - a financial institution
  "bank" - the margin of a river

Morphological decomposition

Morphological decomposition: decomposition of compound words into their word-forming elements, called morphemes

- 'Abfallbeseitigung' ⇒ 'Abfall' and 'Beseitigung'
- 'Krankenhausbibliothek' ⇒ 'Krankenhaus' and 'Bibliothek'

Sometimes decomposition does not make sense:

- 'Eisenbahn', 'Handschuh'

Semantic decomposition: decomposition into conceptual units that reproduce initial concept

- 'Eisenbahn' ⇒ 'Schienenverkehr' and 'Überlandverkehr'
- 'Handschuh' ⇒ 'Hand' and 'Bekleidung'

Thesaurus

[Kom75] defines a thesaurus as follows:

A thesaurus is a natural-language-based documentation language that aims to provide a reversibly unambiguous mapping of natural-language terms and labels by exercising complete vocabulary control and terminological control, and by representing terms and relations between them through the representation of relations between labels and, where appropriate, additional aids.

Descriptor

Calvin Mooers introduced the concept of a descriptor in 1956:

- Representative of a word family or term
- Establishing the meaning by definition
- Validity only within one system

Thesaurus relations

- Thesaurus relation: relation between words in a thesaurus
- Three basic types according to DIN 1643-1:
  1. equivalence relation: relations between real synonyms
  2. hierarchy relation:
     1. abstraction relation (generic relation, e.g. 'truck is a motor vehicle')
     2. composition relation (partitive relation, e.g. 'a car consists of engine, ...')
  3. association relation: denote related terms, such as 'SQL' and 'databases'

Thesaurus relations

Relations to be managed in a thesaurus, if one does not want to distinguish between abstraction and composition relations

Abbreviation	**Meaning**
TT	top term (head term of a hierarchy)
BT	superordinate term
NT	subordinate term
RT	related term (association relation)

B for broader, N for narrower, T for term

Thesaurus relations

Division of the relations BT and NT, if one wants to distinguish between abstraction and composition relation

Abbreviation	**Meaning**
BTG	generic term (abstraction relation)
NTG	subterm (abstraction relation)
BTP	association term (composition relation)
NTP	partial term (composition relation)

B for broader, N for narrower, T for term, G for generalization, P for participation

Use of a thesaurus in an IR system

- Query extension
  $\Rightarrow$ higher recall value, worse precision value
  $\Rightarrow$ should be optional for users
- Creation of document representation
  Use of the preferred terms
  $\Rightarrow$ reduced memory requirements
  $\Rightarrow$ higher recall value, possibly worse precision value

3.6 A Language-independent Approach

- Problem with approaches to *base and stem form reduction, to compound word consideration*: dependence from certain language
  $\Rightarrow$ new language requires adaptation
- Language-independent approach: n-grams
  - trigrams (n = 3): Eisenbahn $\Rightarrow$ eis, ise, sen, enb, nba, bah, ahn
  - bigrams (n = 2): Eisenbahn $\Rightarrow$ ei, is, se, en, nb, ba, ah, hn
  - sometimes with word limit:
    Eisenbahn $\Rightarrow$ #ei, is, se, en, nb, ba, ah, hn#

Evaluation of n-grams

- n-grams for indexing instead of words
- 3 principal approaches to the vector space model
  1. words as terms (= dimensions of the vector space)
  2. stem forms as terms
  3. 5-grams as terms
     Steps: Remove punctuation, convert to lowercase, map numbers to single characters

Evaluation of n-grams

- TREC Investigation:
  - short queries $\Rightarrow$ words and stem forms as terms better than 5-grams
  - long queries $\Rightarrow$ 5-grams better
- Problem: explanation of search result based on n-grams is impossible

3.7 Marking by means of Metadata

- Reducing the vagueness of natural language using metadata
- Special fields for documents such as for *title*, *author*, *creation date*
- Examples:
  - Dublin Core
  - Semantic Web

Dublin Core

- Dublin Core Metadata Initiative (DCMI): Founded in 1995 in Dublin, Ohio at workshop organized by *Online Computer Library Center* (OCLC) and the *National Center for Supercomputing Applications* (NCSA)
- Goal:
  - development of a standardized metadata set for the description of digital documents
  - commonly understood semantics and extensible
  - creation of metadata by authors

Marking by means of metadata

- The core set of Dublin Core: 15 elements
- Elements are optional and can also be listed repeatedly

Dublin Core Element	Use	Possible Data Value Standards
Title	A name given to the resource.	
Subject	The topic of the resource.	Library of Congress Subject Headings (LCSH)
Description	An account of the resource.	
Creator	An entity primarily responsible for making the resource.	Library of Congress Name Authority File (LCNAF)
Publisher	An entity responsible for making the resource available.	
Contributor	An entity responsible for making contributions to the resource.	Library of Congress Name Authority File (LCNAF)
Date	A point or period of time associated with an event in the lifecycle of the resource.	W3CDTF
Type	The nature or genre of the resource.	DCMI Type Vocabulary
Format	The file format, physical medium, or dimensions of the resource.	Internet Media Types (MIME)
Identifier	An unambiguous reference to the resource within a given context.	
Source	A related resource from which the described resource is derived.	
Language	A language of the resource.	ISO 639
Relation	A related resource.	
Coverage	The spatial or temporal topic of the resource, the spatial applicability of the resource, or the jurisdiction under which the resource is relevant.	Thesaurus of Geographic Names (TGN)
Rights	Information about rights held in and over the resource.	

https://guides.library.ucsc.edu/c.php?g=618773&p=4306386

Marking by means of metadata

Language	The language(s) of the intellectual content of the resource is/are noted here.
Relation	The information in this field allows to show connections between different resources that have a formal relation to each other, but exist as independent resources. Examples are images in a document, chapters of a book, or individual items in a collection.
Coverage	Here, information on spatial determination (e.g. geographic coordinates) and temporal validity are entered, which characterize the resource.
Rights	Intended for the content of this element is a link (e.g., via a URL) to a copyright notice, a rights management notice about the legal terms, or possibly a link to a server that dynamically generates such information.

Marking by means of metadata

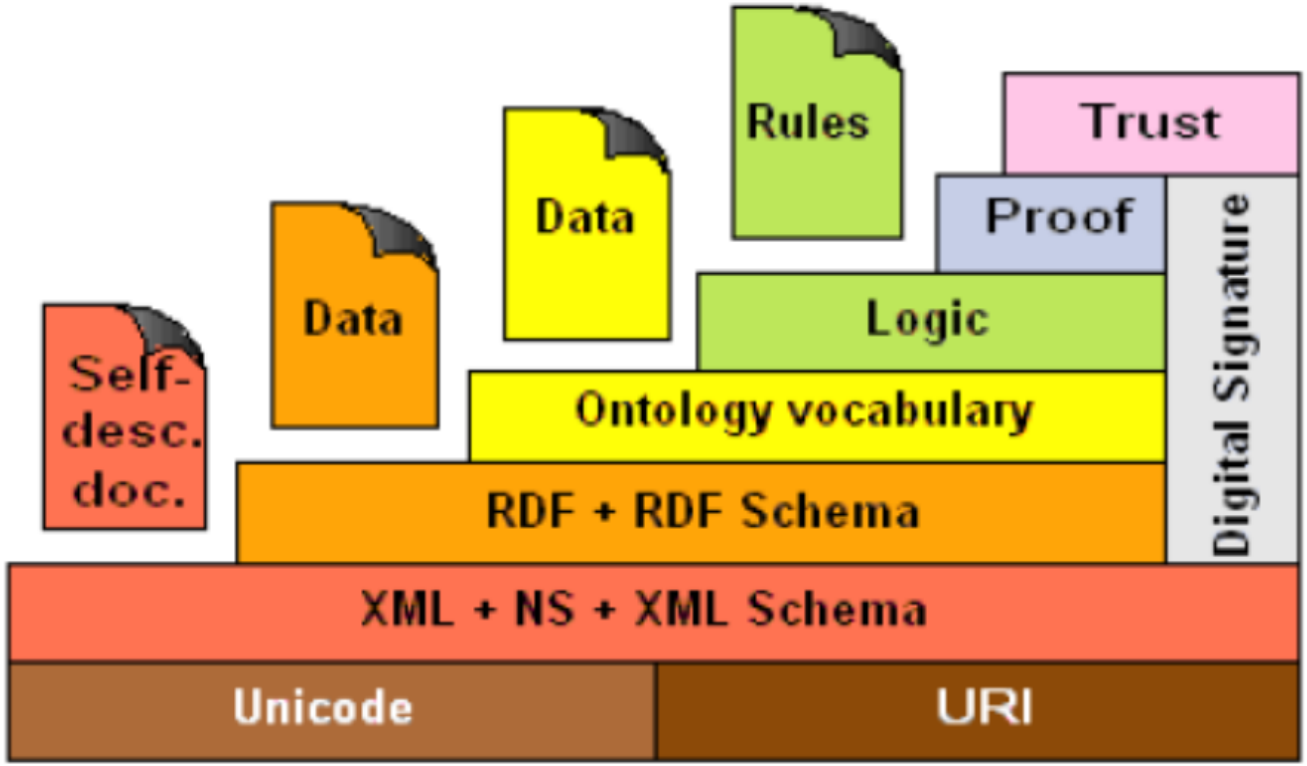
Specification, for example, in the head block of an HTML file:

```
<html>
  <head>
    <title>A Dirge</title>
    <link rel="schema.DC"
          href="http://purl.org/DC/elements/1.0/">
    <meta name="DC.Title"     content="A␣Dirge">
    <meta name="DC.Creator"   content="Shelley,␣Percy␣
          Bysshe">
    <meta name="DC.Type"      content="poem">
    <meta name="DC.Date"      content="1820">
    <meta name="DC.Format"    content="text/html">
    <meta name="DC.Language"  content="en">
  </head>
```

Marking by means of metadata
Semantic Web

- Term "Semantic Web" by Tim Berners-Lee
- Goal: extend WWW to better capture document semantics through algorithms
  $\Rightarrow$ better search, better document exchange
- Realization:
  - content enrichment using machine-understandable metadata
  - layer architecture
- Standardization by *World Wide Web Consortium* (W3C)

Marking by means of metadata
Semantic Web



Marking by means of metadata
Semantic Web

- Unicode: standard with characters for many languages
- URI (*Uniform Resource Identification*): uniform addressing
- RDF (*Resource Description Framework*): representation of metadata based on XML
- Ontology:
  - relationship/rules between individual elements
  - W3C: OWL (*Web Ontology Language*)
- Logic: machine reasoning on metadata
- Trust and proof: establishing trustworthiness by means of signatures and encryption

RDF model

- General language for metadata for WWW resources
- Resources: documents, images, any objects such as goods with price and availability
- Example: *the website* `http://ai1.inf.uni-bayreuth.de/` *has an author whose name is Andreas Henrich*
  - URL for identifying a resource ⇒ *Subject*
  - *Author* as property (attribute) ⇒ *Predicate*
  - *Andreas Henrich* as attribute value ⇒ *Object*

RDF model

Conversion of natural language statements into machine-understandable format required:

- machine-understandable identifiers for subject, predicate and object
- machine-readable format for identifiers

$\Rightarrow$ Uniform Resource Identifier (URI)

Uniform Resource Identifier (URI)

- URL for web pages as subset of URI
- URI for unique identification of any objects
- RDF: URI for subject, predicate and object

RDF model

- Example: *the website* `http://ai1.inf.uni-bayreuth.de/` *has an author whose name is Andreas Henrich*

- Components identified by URI
  - Subject:
    `http://ai1.inf.uni-bayreuth.de/`
    ⇒ URL
  - Predicate:
    `http://purl.org/dc/elements/1.1/creator`
    ⇒ Use of Dublin Core
  - Object:
    `http://ai1.inf.uni-bayreuth.de/mitarbeiter/henrich/`
    ⇒ Homepage of Andreas Henrich

RDF model as graph

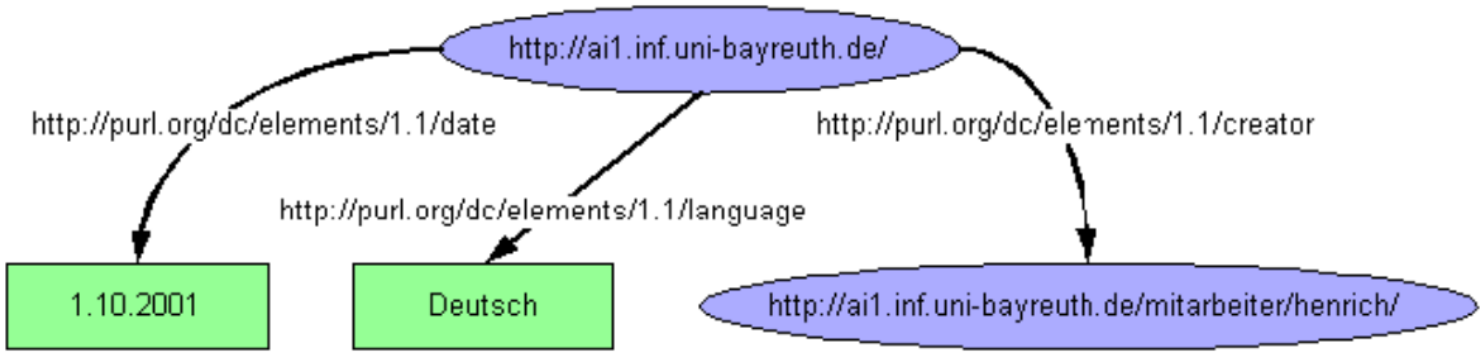
RDF statements as nodes and edges of a graph

- node: subject or object
- named and directed edge: predicate

RDF model as graph

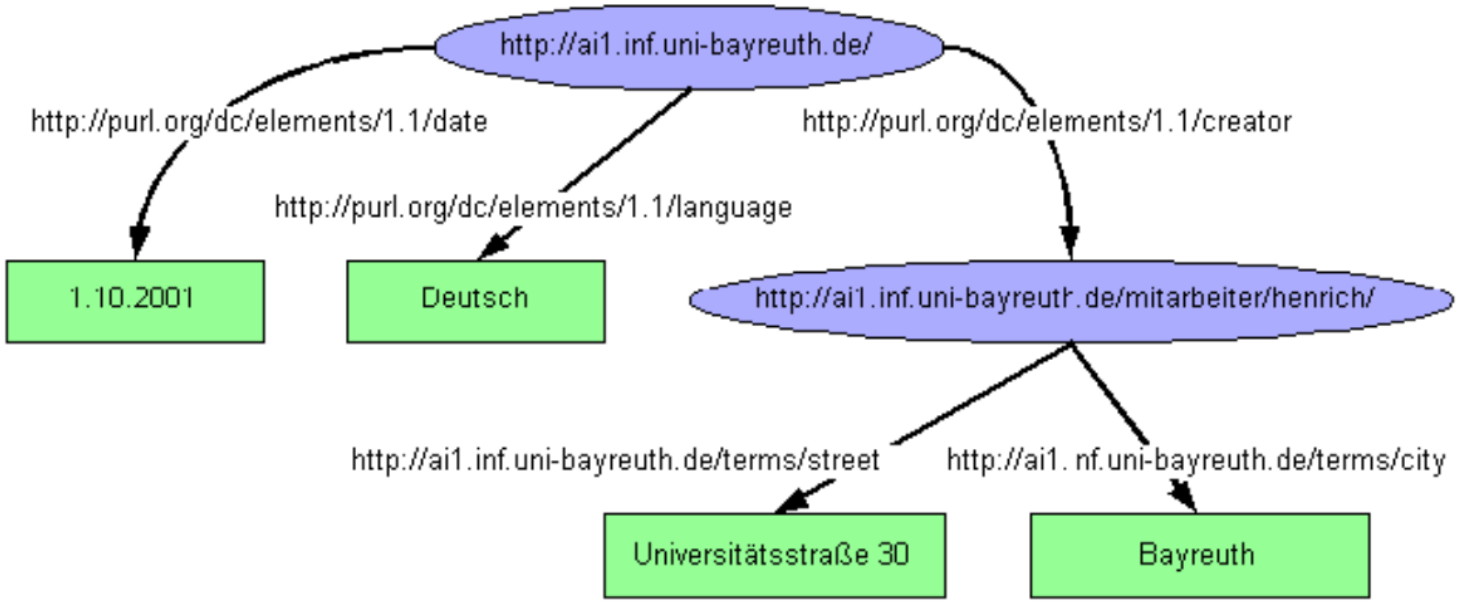
- Collection of statements, such as
  - Website http://ai1.inf.uni-bayreuth.de/ has author whose name is Andreas Henrich
  - Website http://ai1.inf.uni-bayreuth.de/ has creation date with value 1.10.2001
  - Website http://ai1.inf.uni-bayreuth.de/ has a language value of German
- Common graph of nodes and edges

RDF model as graph



- Value as literal and not as URI
- Object of a statement can be subject of another statement

RDF model as graph



- Additional metadata is given for the author (*street*, *city*)
- Author is therefore simultaneously object of one statement and subject of the second statement

RDF/XML Syntax Specification

- RDF defines a syntax for mapping RDF graphs into XML
- Nodes and edges of an RDF graph are mapped to elements, attributes, element contents, and attribute values in XML
- URIs for properties
- Objects are mapped using namespaces

RDF/XML Syntax Specification

Example:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<rdf:RDF xmlns:rdf="http://www.w3c.org/1999/02/22-rdf-
    syntax-ns#" xmlns:dc="http://purl.org/dc/elements/1.1/"
    >

  <rdf:Description rdf:about="http://ai1.inf.uni-bayreuth.
      de/">
    <dc:date>1.10.2001</dc:date>
    <dc:language>Deutsch</dc:language>
    <dc:creator rdf:resource="http://ai1.inf.uni-bayreuth
        .de/mitarbeiter/Henrich/"/>
  </rdf:Description>
</rdf:RDF>
```

- Predicate as element from Dublin Core model

RDF/XML Syntax Specification

Extension with locally defined vocabulary *ex*

```
<?xml version="1.0" encoding="ISO-8859-1"?>
  <rdf:RDF xmlns:rdf="http://www.w3c.org/1999/02/22-rdf-
      syntax-ns#" xmlns:dc="http://purl.org/dc/elements
      /1.1/" xmlns:ex="http://ai1.inf.uni-bayreuth.de/
      terms/">

    <rdf:Description rdf:about="http://ai1.inf.uni-
        bayreuth.de/">
      <dc:date>1.10.2001</dc:date>
      <dc:language>Deutsch</dc:language>
      <dc:creator rdf:resource="http://ai1.inf.uni-
          bayreuth.de/mitarbeiter/Henrich/">
        <ex:street>Universitätsstraße 30</ex:street>
        <ex:city>Bayreuth</ex:city>
      </dc:creator>
    </rdf:Description>
  </rdf:RDF>
```

RDF Schema

- A type system can be created by means of RDF Schema (RDFS)
- Classes:
  - similar to classes in object-oriented programming languages
  - can represent any objects, such as web pages, people, document types, things, or even abstract concepts
  - inheritance
- RDF Schema is created in RDF

RDF Schema: Classes

In the following, three classes are defined in RDFS:

- *Automobil*: base class *Automobil* is derived from RDF Schema base class *Resource*
- *Pkw*: class *Pkw* is derived from class *Automobil*
- *Sportwagen*: the class *Sportwagen* in turn from class *Pkw*

RDF Schema: Classes

```
<rdf:RDF xmlns:rdf="http://www.w3c.org/1999/02/22-rdf-
    syntax-ns#" xmlns:rdfs="http://www.w3c.org/2000/01/rdf-
    schema#">

  <rdf:Description rdf:ID="Automobil">
    <rdf:type rdf:resource="http://www.w3c.org/2000/01/
        rdf-schema#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.w3c.org
        /2000/01/rdf-schema#Resource"/>
  </rdf:Description>

  <rdf:Description rdf:ID="Pkw">
    <rdf:type rdf:resource="http://www.w3c.org/2000/01/
        rdf-schema#Class"/>
    <rdfs:subClassOf rdf:resource="#Automobil"/>
  </rdf:Description>

  <rdf:Description rdf:ID="Sportwagen">
    <rdf:type rdf:resource="http://www.w3c.org/2000/01/
        rdf-schema#Class"/>
    <rdfs:subClassOf rdf:resource="#Pkw"/>
  </rdf:Description>
</rdf:RDF>
```

RDF Schema: Properties

Properties (predicates) for the individual classes can be defined as follows:

```
<rdf:RDF xmlns:rdf="http://www.w3c.org/1999/02/22-rdf-
    syntax-ns#" xmlns:rdfs="http://www.w3c.org/2000/01/rdf-
    schema#">

  <rdf:Description rdf:ID="zugelassenAuf">
    <rdf:type rdf:resource="http://www.w3c.org
        /1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:domain rdf:resource="#Automobil"/>
    <rdfs:range rdf:resource="http://ai1.inf.uni-bayreuth
        .de/classes#Person"/>
  </rdf:Description>

  <rdf:Description rdf:ID="MaxAnzahlPassagiere">
    <rdf:type rdf:resource="http://www.w3c.org
        /1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:domain rdf:resource="#Pkw"/>
    <rdfs:range rdf:resource="http://www.w3c.org/2001/
        XMLSchema#integer"/>
  </rdf:Description>

</rdf:RDF>
```

RDF Schema

- Properties (predicate) as edge have input type (domain) and target type (range)
  - input type:
    - must be a class
  - target type:
    - can be a literal
    - but also a class

RDF Document

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3c.org/1999/02/22-rdf-
    syntax-ns#" xmlns:a="http://ai1.inf.uni-bayreuth.de/
    schemas/autos#">

  <a:Pkw rdf:ID="AndreasHenrichsPkw">
    <a:zugelassenAuf rdf:resource="http://ai1.inf.uni-
        bayreuth.de/mitarbeiter/henrich/"/>
    <a:MaxAnzahlPassagiere rdf:datatype="http://www.w3c.
        org/2001/XMLSchema#integer">5</
        a:MaxAnzahlPassagiere>
  </a:Pkw>
</rdf:RDF>
```

Here subject with *rdf:ID* instead of *rdf:about* $\Rightarrow$ referencable via

- *Relative URI:*
  "#AndreasHenrichPKW"
- *Absolute URI:*
  "http://ai1.inf.uni-bayreuth.de/parkplatz/#AndreasHenrichPKW"