Space, Time, and Semantics

Willem Robert van Hage
Vrije Universiteit Amsterdam
wrvhage@few.vu.nl
tight integration between
Space, Time,
and Semantics
in SWI-Prolog

Willem Robert van Hage
Vrije Universiteit Amsterdam
wrvhage@few.vu.nl
The semantic web is a web of data. As with the “regular” web, there is no centralized scheme. Data are shared along shared concepts.
DBpedia

umbel-sc:Person
... 

umbel-sc:Municipality
... 

db:Amy Koopmanschap
dbp:name = "Amy Koopmanschap"
dbp:party = dbp:GL
...

dbp:leaderName
db:Diemen
rdfs:label = "Diemen"
dbo:populationTotal = 23866
...

db:Diemen
owl:sameAs

db:Amy Koopmanschap
dbp:leaderName
rdfs:subClassOf
umbel-sc:Person
rdf:type

Geonames

geo:Feature
geo:name
wgs84:lat
wgs84:long
...
geo:parentFeature
geo:Code
geo:featureCode
rdfs:label = "populated place"
geo:P
PPL
geo:featureCode
rdf:type

rdfs:label = "Diemen"
wgs84:lat = 52.33833
wgs84:long = 4.95972
geo:2756888

dbo:populationTotal = 23866
linking open data
the web as a database

- you can see the semantic web as a world-wide database
- every edge in the cloud of linked data sets is a database “join” over a common concept
the web as a database

- database tables are indexed along key properties that identify records,
- combination of data happens by joining on these keys
- keys are selected to optimize certain important types of queries
- on the web, it is not known in advance which properties are useful to use as keys to combine data
the web as a database

- on the web anything is potentially interesting, so everything has to be indexed
- everybody hosts part of the web and can decide if and how to index its resources
  - RDF(S) and OWL are used as exchange formats, not necessarily for internal representation
- some hosts will deal with large sets of instances, others with complex ontologies, yet others will be GISs
anytime results

• on the web you can not do a full “join” of everything you like to reason about

• the data are all over the web, incomplete, and do not fit on a single machine

• you need to decide how to traverse the enormous search space
SWI-Prolog

• to search in the large triple space we use SWI-Prolog’s semweb package

• Prolog offers an elegant way to control the search strategy through logical statements

  amsterdam_artist(Name) :-
    rdfs_instance_of(Who,'Artist'),
    rdf(Who, dbp:birthPlace, Where),
    rdf(Where, geo:name, literal('Amsterdam')),
    rdf(Who, dbp:name, literal(Name)).

• Prolog tries to instantiate all query variables one by one in the order they are needed
indexing in Prolog

• normal Prolog clauses are indexed linearly by their arguments e.g. `functor1(arg2, arg3, arg4)`.

• RDFS and partial OWL reasoning is implemented with Prolog rules

• RDF triples are hashed for fast lookup e.g. `rdf(Subject, Predicate, Object)` is put in a S, P, O, SP, SO, OP, and SPO hash for fast existence testing

• literals are indexed in a prefix B-tree for fast substring and prefix search
complex queries

• the type of problems we deal with requires a combination of all of the above and more
  • subsumption reasoning
  • substring lookup
  • spatial containment and nearest-neighbor

• “give me artists that were born near Amsterdam”
  • subsumption reasoning
  • ordered results
  • lexical comparison
  • nearest neighbor
e.g. painters, actors, writers

• “which works were made in the same period”
  • containment
indexing in Prolog

• out of the box, SWI-Prolog can not do multidimensional indexing
e.g. rdf(db:'Amsterdam', georss:point, literal("52.371 4.897")).

• that makes searching for the nearest or contained objects in space (or time) very slow

• querying a remote GIS for incremental nearest-neighbor is also very slow (latency, loss of state)
the space package

- an extension of SWI-Prolog
- written in C++, wraps the “spatialindex” library by Marios Hadjieleftheriou
- supports non-deterministic incremental nearest neighbor [Hjaltason & Samet 1999], containment, and intersection queries on R-trees [Guttman 1984]
- will support TPR-trees [Tao & Sun 2003] and MVR-trees [Nakamura & Dekihara 2003], points, polygons, and circles
how does it work?

• the spatial search terminates after each result, storing its state to continue later

• it only continues to search (where it stopped) when more results are needed

• this allows fine grained interaction between spatial and other types of criteria, which leads to good performance
?- nearest(point(52.34,4.96), N).
indexing...
N = 'http://sws.geonames.org/2756888/' ;
N = 'http://sws.geonames.org/6639950/' ;
N = 'http://sws.geonames.org/6639949/' ;
N = 'http://dbpedia.org/resource/Diemen' ;
N = 'http://sws.geonames.org/2749167/' ;
...

?- nearest(point(52.34,4.96), N),
    rdf(N, geo:name, literal(L)).
N = 'http://sws.geonames.org/2756888/',
L = 'Diemen' ;
N = 'http://sws.geonames.org/6639950/',
L = 'Station Diemen' ;
N = 'http://sws.geonames.org/6639949/',
L = 'Station Diemen Zuid' ;
N = 'http://sws.geonames.org/2745507/',
L = 'Vinkebrug' ;
N = 'http://sws.geonames.org/6639767/',
L = 'Station Duivendrecht' ;
N = 'http://sws.geonames.org/6639766/',
L = 'Station Bijlmer Arena'.
...

?- nearest(point(52.34,4.96), N),
    rdf(N, geo:featureClass, geo:'S'),
    rdf(N, geo:name, literal(L)).
N = 'http://sws.geonames.org/6639950/',
L = 'Station Diemen' ;
N = 'http://sws.geonames.org/6639949/',
L = 'Station Diemen Zuid' ;
N = 'http://sws.geonames.org/2745507/',
L = 'Vinkebrug' ;
N = 'http://sws.geonames.org/6639767/',
L = 'Station Duivendrecht' ;
N = 'http://sws.geonames.org/6639766/',
L = 'Station Bijlmer Arena'.
...
my questions

• how do you tie spatial objects to URI’s? (e.g. to annotate objects in shapefiles or to label concepts with shapes)
• how can you interface with spatial UIs? (e.g. to get queries and show query results)
• are there free collections of shapes? (e.g. countries, cities, waterways, seas, etc.)
• which vocabularies do you use?
• what kind of semantic queries do you use?
• who would like to try out this software?
• who would like to explain GML to me in depth and talk about / work on GML-RDF conversion?