Archived Document

Sept. 28, 2005
Glenn Takanishi
Neuralmachines

Notes on Metadata Based Text Search

This report present the elements of a knowledge base system based on metadata objects. Metadata is data describing the data and itself. Extracting meaning from data requires the mechanical data extractor to assume a proper context for the data. Context is acquired by semantic pattern recognition methods such as verb (predicate) ordering, and entity (subject) classifcation. As much as possible, context is acquired indirectly from a hierarchical ontology used as the pattern matching template. Using metadata as object-oriented data structures which can attribute elements of itself to the data and other metadata objects enables the mechanical data extractor to determine the context to a limited extent.

1. Introduction

An atomic[1][1] term used in describing elementary concepts is called the subject-predicate-object, SPO, atom. These terms are interconnected elements in a graph. The subject-objects are the vertices and the predicate or verb is the link in the graph. The W3C calls these elements RDF [2] descriptors which is analogous, in its ontology model, to nodes in a directed graph.

The RDF specification is a set of precise semantics for 3-tuples and the corresponding inference rules for building a system composed of metadata elements called triplets. These SPO triplets have the form analogous to the 3-tuples used in classical KL-One type of programs.[2]

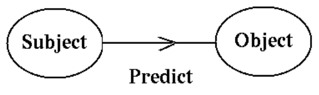

Figure 1: **RDF Triple**

A set of RDF triplets is called an RDF graph.

1 KIF, Knowledge Interchange Format Reference Manual, Version 3, Genesereth and Fikes, June 1992
2 OPS5, Soar, and Loom, Classic

A graph can be organized in a non-hierarchical structure as opposed to a hierarchical btree below.

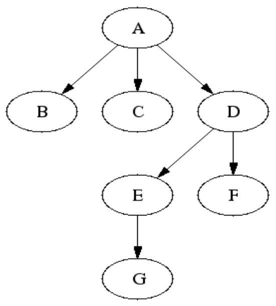

Figure 2: **Btree Graph**

An RDF graph is composed of a set of RDF triples. The direction of the flow of action from subject to object makes the RDF triples in graph language terminology a directed graph.

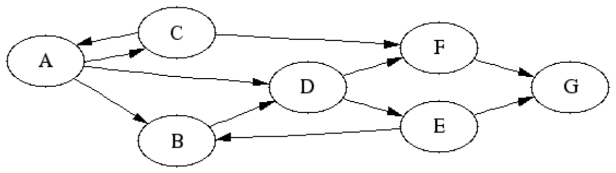

Figure 3: **Directed RDF Graph**

When the number of RDF nodes becomes large as in a representation of a textual document, then the computer resources required to perform pattern matching[3] increases exponentially as a function of the number of nodes, n. With n easily approaching a million nodes for a set of documents, the exact solution to the pattern matching problem becomes intractable. For a semantic language representation of RDF triples on textual data, the representation maybe is inherently ill-defined, and seeking an exact solution probably is not worthwhile.

Symbolic representation of "natural" language is more expressive and powerful then the mapping of textual document elements into static keyword eigenvectors. In fact, without being put in the proper context, vector decomposition of a document set degrades the accuracy of the data representation. In practical applications, the user selects the eigenvector set to guide the clustering process.

3 Pattern matching for Meta is done using PRISM, www.neuralmachines.com. It's augmented using the Bag of Words, BOW, library by Andrew McCallum; http://www.cs.cmu.edu/~mccallum/bow/. "*Bow* (or *libbow*) is a library of C code useful for writing statistical text analysis, language modeling and information retrieval programs. The current distribution includes the library, as well as front-ends for document classification (*rainbow*), document retrieval and document clustering."

One great area I have not worked on is parsing sentences in natural language text. To make RDF records, the mechanical extractor needs to know the parts of the sentence, ie., subject, verb and object. We've started using the Brill POS tagger; http://www.d.umn.edu/~tpederse/pos.html. "In (Brill 1992), a trainable rule based tagger is described that achieved performance comparable to that of stochastic taggers.." -- Eric Brill.

The RDF elements represents rather nebulous attributes of text data. Extracting meaning by performing inexact or approximate pattern matching on related RDF nodes makes the problem easier to compute.

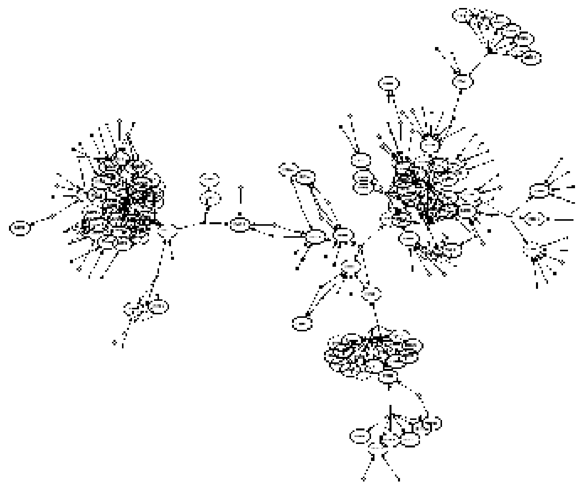

Figure 4: **Directed RDF Graph**[4]

In theory, extracting complex of meaning from a textual dataset requires analyzing the internal structures or linkages of the dataset's semantic network. This requires knowing details of the document structure beyond what is possible by using clustering methods. For example, knowing the direction of flow of the RDF linkage elements, eg., direction of action of a verb, can change the meaning of a document.

A hierarchical ontology contains the structure which the mechanical pattern matcher can use to extract meaning. The more flexible and powerful the ontology is the more meaning can be derived from it. Datasets containing very complex meaning require very flexible ontologies for its representation. Using objects, classes and metaclasses, in hierarchical ontologies is a natural way to increase the mechanical classifier's power of representating concepts. Class systems inherently assume subsumption relationships and indirectly inferes the relationships defined in the hierarchical ontologies.

1.1. Metadata Objects

Currently, core pattern matcher is the Rete (UPS) inference engine. The design of metadata objects using UPS and the POS tagger is underway so that we can derive assertions from metadata.

2. Inference and Graph Networks

The inference mechanisms of the knowledge base are created by organizing the graph into a network of specialized elements. With respect to a semantic network, the elements could be organized by subsuming the subject and object entities under the predicate or action verbs. Learning systems like Soar initially needs to be trained in classifying or subsuming concepts. Concepts are classified according to a knowledge hierarchy or ontology. Without an ontology, most knowledge based systems are not smart enough to build a subsumption infrastructure by themselves.

4 Generated by Graphviz, http://www.graphviz.org/Gallery/undirected/softmaint.html

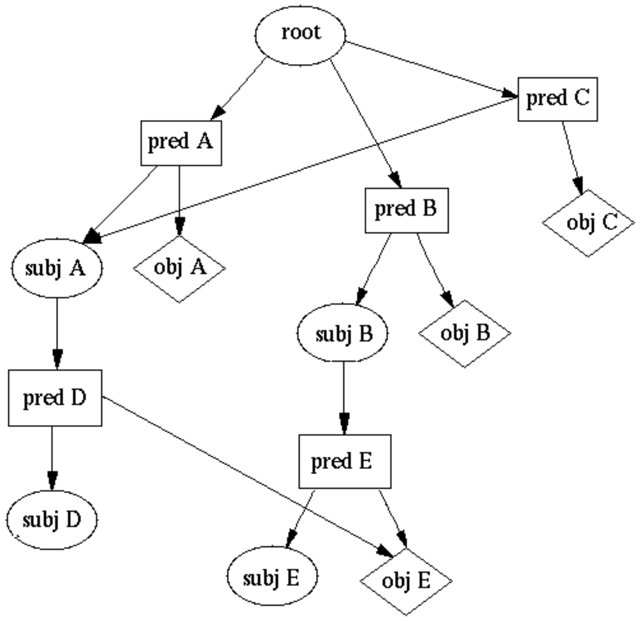

Figure 4: **Directed RDF Graph**[5]

Usually subsumption of triplets is decidable, and easier than inference in full first order logic. However, there are cases when subsumption is undecidable[5]. Restricting the description leads to polynomial time subsumption, however, the description might then not be able to fully express the concepts in the model's simulation. Sometimes definitions of concepts which are natively part of the simplest language description of the model is an NP-hard subsumption problem [6].

In practice, highly expressive descriptions are not usable because the computational complexity of reasoning becomes too high. But experience has shown that moderately expressive descriptions are computationally feasible, and produce operationally correct results.

3. Production Systems

A production system arranges the 3-tuples into a optimal network for performing inferences. The nodal elements of the triplets are arranged so that the inference engine contains the minimum amount of search paths containing the organized nodal elements (figure 4). These nodal elements are called working memory elements, WME's, in Rete algorithm terminology. WME''s are triplets. For example, (subj A, pred A, obj A).

A singular production is an if-then clause operation. In analogy to a database, a singular production can be compared to a query. The test in a condition represents a SELECT operation over a single WME relation. Tests run over a single WME relationship is by convention in Rete terminology called an alpha memory test. By convention, tests including multiple WME's are called beta memory tests [7: page 10-11]. The Rete algorithm was designed to run productions simulating many queries simultaneously.

The production system we're using is called UPS. It's the version on the CMU archive was developed by

5 Nodes elements in a Rete tree; inside UPS [7].

Dirk Kalp and Anurag Acharya than with CMU in the late 1980's. This software uses the Rete Algorithm in our RDF storage base systems. For example, the application UPS-2, uses the Rete based networks to perform inferencing.

Rules submitted to UPS are used to reduce the number of steps required to traverse the search space of WME triples. The Rete algorithm “prunes” the search space by making the directed Rete graph optimal or most efficient for navigation. In the Rete algorithm, this results in a nealy linearly slowdown of computational time verses other systems which require a higher order (O2 or exponential) computational time as the pattern matching cost increases.

In daily practice, we fight this slowdown problem by seeking to improve every element in the match process. The better the programmer formulates the rulesets, the quicker and more accurately inferences will be made.

The application search space can be reduced independently of Rete's pattern matching algorithms. This requires careful analysis of the data contexts and appropriate construction of rulesets. This strategy attempts to reduce the match cost for each individual, taken one at a time[7]. Limiting the number of rules used by making each rule as general as possible without reducing a rule's power to resolve specific instances is an optimization strategy which lowers match cost. The programmer intuitively knows that a large number of cheap rules greatly increases the match cost while degrading inference resolution.

The following shows 3-tuple input into a Rete working memory element, WME[7].

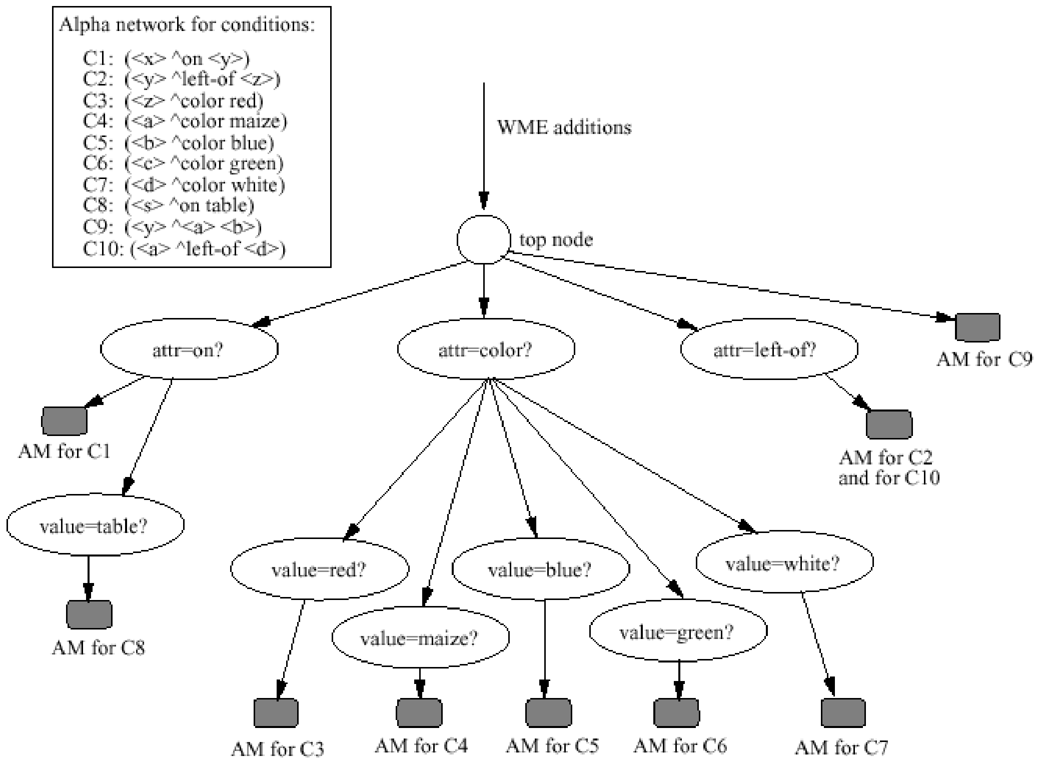

Figure 5: **Dataflow into Alpha WME [7: page 14]**

3.1. The Metadata Extractor, Meta

The metadata creator is the stepstone of our software tools. It's required to create the RDF records from reading the textual datasets. Meta uses the hierarchical ontologies to perform pattern matching. It culls the keywords down the ontology chains into profiles which combined appropriately generates different 'views" of the dataset.

Meta is required to read text datasets efficiently with a set of 10,000 ontology profiles containing more than a total of 100,00 keywords. I'm rather certain meta can read text data, and create RDF records, at a rate of 10 pages per second (on average about one document or message per second.)

4. RDF Query

Quote[9]:

“SPARQL is a query language for accessing such RDF graphs. It provides facilities to:

- extract information in the form of URIs, bNodes, plain and typed literals.

- extract RDF subgraphs.
- construct new RDF graphs based on information in the queried graphs. "

I haven't used SPARQL yet.

Summary

So far, I haven't developed a reasonable rulesets for our application using UPS. I've been working on the metadata extractor.[6]

I'm trying to keep everything as simple as possible. This is hard because most of the software tools are very comprehensive requiring lots of time. Approximate graph pattern matching, and clustering using eigenvector decomposition can always be greatly aided by good RDF data extracted with parsers using first order logic.

RDF, at the least is simple. You can create amazing complex graphs by throwing enough nodes together.

The next update to these notes will be on the progress of Meta, the RDF data extractor.

6. References

1. Semantic Memory, Ross Quinlan, 1968, Semantic Information Processing, M. Minsky (ed), 216-260, MIT Press

2. RDF Semantics: W3C Recommendation 10 February 2004; http://www.w3.org/TR/2004/REC-rdf-mt-20040210/

3. UPS, the Unofficial Production System

4. IBID.

5. Subsumption in KL-One is Undecidable;Manfred Schmidt-Schauss, Goethe University, 1989

6. Terminological Reasoning is Inherently Intractable; Berhnard Nebel, Artificial Intelligence, 43: 235-249, 1990

7. Production Matching for Large Learning Systems; Robert Doorenbos, CMU-CS-95-113, 1995

8. OWLJessKB: A Semantic Web Reasoning Tool, http://edge.cs.drexel.edu/assemblies/software/owljesskb/

9. SPARQL Query Language for RDF, W3C Working Draft 12 October 2004, http://www.w3.org/TR/2004/WD-rdf-sparql-query-20041012/

6 The backend for Meta which writes the RDF XML output uses PERL scripts, Redland's RAPTOR and the libxml2 library.