Future Directions For Semantic Systems

John F. Sowa

20 August 2010

The Challenge

Theorem provers in the 1960s and ’70s could perform deduction faster and more accurately than most people.

But in everyday reasoning, people use background knowledge that computers don’t have.

Today, computers have vast amounts of data, but they can’t interpret it as knowledge.

What kinds of tools could enable computer systems to

- Collaborate with people in order to analyze, organize, and interpret the data as knowledge?
- Help people use the knowledge in more effective ways of reasoning, planning, and problem solving?

The Knowledge Acquisition Bottleneck

Knowledge representation requires training in logic, ontology, conceptual analysis, and system design.

Annotating large volumes of data with semantic tags

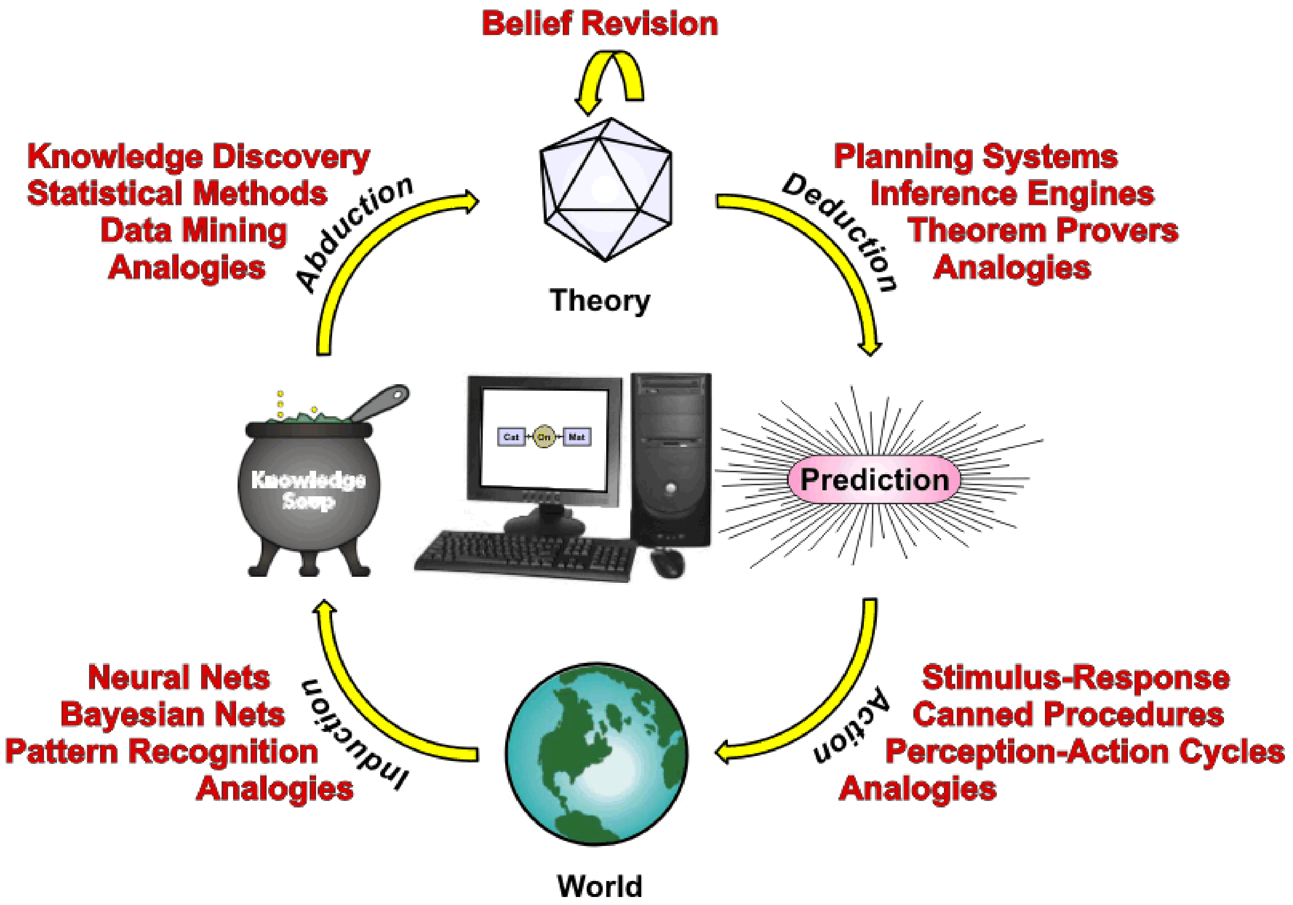
- Requires somewhat less training,
- But different annotators frequently disagree,
- And annotation is tedious, error-prone, and expensive.

Training statistical tools on a “Gold Standard” requires

- Highly paid experts to create the gold standard,
- A new gold standard for every subject, genre, and style.

Can we develop tools that more people can use more easily without a lengthy and costly amount of training?

Peirce’s Cycle of Reasoning



Belief Revision

Knowledge Discovery
Statistical Methods
Data Mining
Analogies

Abduction

Theory

Deduction

Planning Systems
Inference Engines
Theorem Provers
Analogies

Prediction

Action

Stimulus-Response
Canned Procedures
Perception-Action Cycles
Analogies

World

Induction

Neural Nets
Bayesian Nets
Pattern Recognition
Analogies

There are many different ways of using knowledge.
New kinds of useful tools are constantly being invented.
We need to integrate those tools with the reasoning cycle.

Case Study #1: Cyc Project

Started in 1984 by Doug Lenat.

Name comes from the stressed syllable of 'encyclopedia'.

Goal: Implement a computable version of the background knowledge shared by most high-school graduates.

After the first 25 years:

- 100 million dollars and 1000 person-years of work,
- 600,000 concepts,
- Defined by 5,000,000 axioms,
- Organized in 6,000 microtheories.

The OpenCyc Foundation made the Cyc ontology and some applications available in open source.

To browse the ontology or to download OpenCyc, see http://opencyc.org/

Focus on Applications

The Cyc ontology is the world's largest body of knowledge represented in logic and suitable for detailed deduction.

The CycL language is a superset of many different versions of logic, including RDF, OWL, and rule-based systems.

Starting in 2004, Cycorp put more emphasis on applications.

Cycorp earned more money from applications in the years 2008 to 2010 than in the previous 24 years.

Some of the fastest growing applications are to medical informatics.

At the Cleveland Clinic, about 1700 axioms from the general Cyc ontology are used to understand and respond to a typical query.

For white papers and research publications about Cyc, see http://cyc.com

Using Cyc as a Development Environment

Cyc is a good platform for defining semantics, and their huge knowledge base is a valuable resource.

Automated tools can extract axioms from Cyc and convert them to the formats used by other kinds of systems.

In fact, some Cyc users developed a “knowledge bus” for extracting axioms and tailoring them to other platforms. *

Such techniques can be used to integrate Cyc with the tools and methodologies used in mainstream IT.

*** Peterson, Brian J., William A. Andersen, & Joshua Engel (1998) Knowledge bus: generating application-focused databases from large ontologies, http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-10/**

Lessons Learned

The first 25 years of research on the Cyc Project developed a huge knowledge base and supporting software.

But the academic research was not easy to commercialize.

Two successful ways of using the Cyc resources:

1. Building bridges from Cyc to other systems (Cleveland Clinic).
2. Mapping knowledge from Cyc to other formats (Knowledge Bus).

Both methods are useful, but they require more skills than conventional programming methodologies.

We need tools and methodologies to enable subject-matter experts (SMEs) to update a knowlede base by themselves.

Case Study #2: Tesco.com

A large UK retailer, Tesco.com, sells a variety of goods, ranging from groceries to electronic equipment.

They wanted a flexible system that would allow employees to update the system dynamically.

One vendor designed a system based on RDF and OWL, but Tesco employees could not modify it.

Calling an OWL expert for every update is too slow, and hiring an expert for every store would cost too much.

They needed a simpler system that current employees could use without lengthy and costly training.

Automated tools must ensure that the rules are consistent and easy for Tesco employees to update and correct.

A Better Alternative

Gerard Ellis, who had over ten years of R & D experience with conceptual graphs, designed and implemented a new system:

- The internal knowledge representation was conceptual graphs.
- The interface for Tesco employees was controlled English.
- Tesco employees could extend or modify the rule base by typing sentences in controlled English.
- The system used the methodology of ripple-down rules to update the knowledge base, check for errors, and preserve consistency.

This application was successfully used for groceries.

It was also adapted to the electrical and wine departments.

Tesco employees could update it with very little training.

See Qusai Sarraf and Gerard Ellis, “Business Rules in Retail: The Tesco.com Story,” *Business Rules Journal*, Vol. 7, No. 6 (Jun. 2006), http://www.BRCommunity.com/a2006/n014.html

Typical Business Rules

Tesco employees typed information in controlled English, from which the system automatically generated the following rules:

- If a television product description contains “28-inch screen”, add a screen_size attribute_inches with a value of 28.
- a) If a recipe ingredient contains butter, suggest “Gold Butter” as an ingredient to add to the basket. b) If the customer prefers organic dairy products, suggest “Organic Butter” as an ingredient to add to the basket.
- If a customer buys 2 boxes of biscuits, the customer gets one free.
- If the basket value is over £100, delivery is free.
- If the customer is a family with children, suggest “Buy one family sized pizza and get one free”.

These rules were automatically generated from a decision tree, as described on the next slide.

Ripple-Down Rules (RDR)

Automated tools with a methodology for building, updating, and maintaining a rule base that is guaranteed to be consistent:

- **Internally, the rules are organized as a decision tree.**
- **Each link of the tree is labeled with one condition.**
- **Each leaf (end point) is labeled with a conclusion (or a conjunction of two or more conclusions).**
- **Any update that would create an inconsistency is blocked.**
- **If the update is consistent, the tree is automatically reorganized.**
- **For maximum performance, the decision tree can be compiled to a nest of if-then-else statements in a programming language.**

RDR is used for medical applications with thousands of rules created by physicians who have no training in AI.

See B. R. Gaines and P. Compton, Induction of Ripple-Down Rules Applied to Modeling Large Databases, http://pages.cpsc.ucalgary.ca/~gaines/reports/ML/JIIS95/index.html

Lessons Learned

Three technologies with complementary strengths:

- Controlled English: Readable by anyone who can read English and easier to write than most computer notations.
- Ripple-down rules: Consistent knowledge bases with thousands of rules can be developed by SMEs with no training in logic or ontology.
- Conceptual graphs: A dialect of Common Logic, which can serve as an intermediate notation between CNLs and other formalisms.

A good combination, but more R & D would be useful:

- The system could be adapted to multiple departments at Tesco.
- But major revisions are necessary for similar applications at other companies or for different kinds of applications at Tesco.
- New development tools should enable IT professionals with little or no training in AI technologies to make such revisions.

Case Study #3: VivoMind

Foundational technologies:

1. VivoMind Analogy Engine (VAE)
2. Flexible Modular Framework (FMF)
3. Prolog for Intelligent Knowledge Systems (PrologIKS)
4. Societies of Heterogeneous Agents
5. VivoMind Language Processor (VLP)
6. Proto-Ontology Extractor
7. Common Logic Controlled English (CLCE)

Assembling these components into an intelligent system requires a considerable amount of expertise.

But the resulting system can be easy to use by SMEs who have never studied logic or ontology.

Four Views of Analogy

1. By logicians:

   Deduction is reasoning from “first principles.”

2. By psychologists:

   Analogy is a fundamental mechanism of human and animal cognition.

   All aspects of language understanding depend on analogy.

3. Theoretical:

   Analogy is a general pattern-matching mechanism, and all methods of formal logic — deduction, induction, and abduction — are special cases.

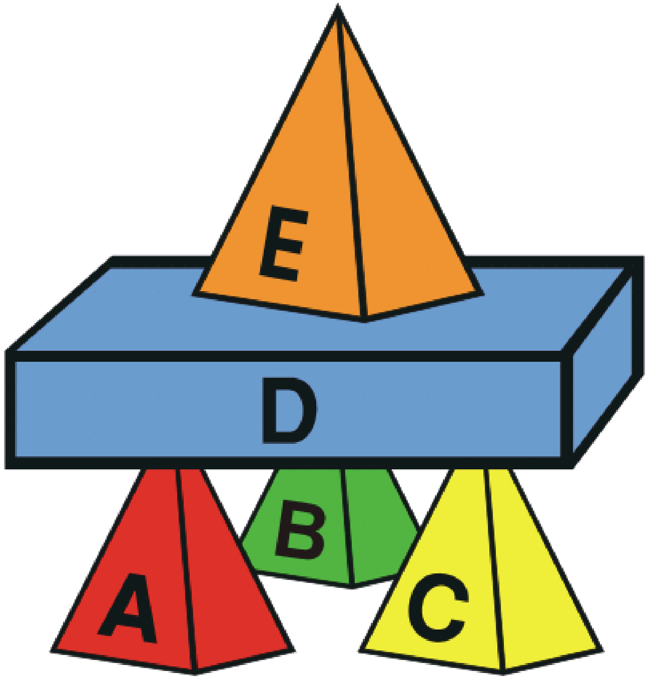
4. Computational:

   A powerful and flexible technique that can have important applications in reasoning, learning, and language processing.

   But practicality depends on finding analogies efficiently.

Describing Things in Different Ways

How can we describe what we see?



In ordinary language?

In some version of logic?

In a relational database?

In the Semantic Web?

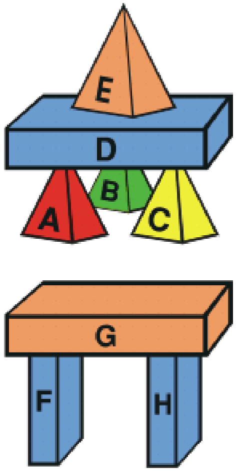
In a programming language?

Even when people use the same language,
they use different words and expressions.

How could humans or computers relate
different descriptions to one another?

Structured and Unstructured Representations

A description in tables of a relational database:



Objects

Entity	Shape	Color
A	pyramid	red
B	pyramid	green
C	pyramid	yellow
D	block	blue
E	pyramid	orange
F	block	blue
G	block	orange
H	block	blue

Supports

Supporter	Supportee
A	D
B	D
C	D
D	E
F	G
H	G

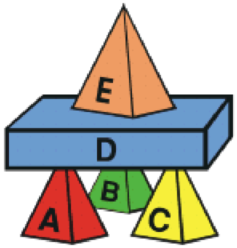
A description in English:

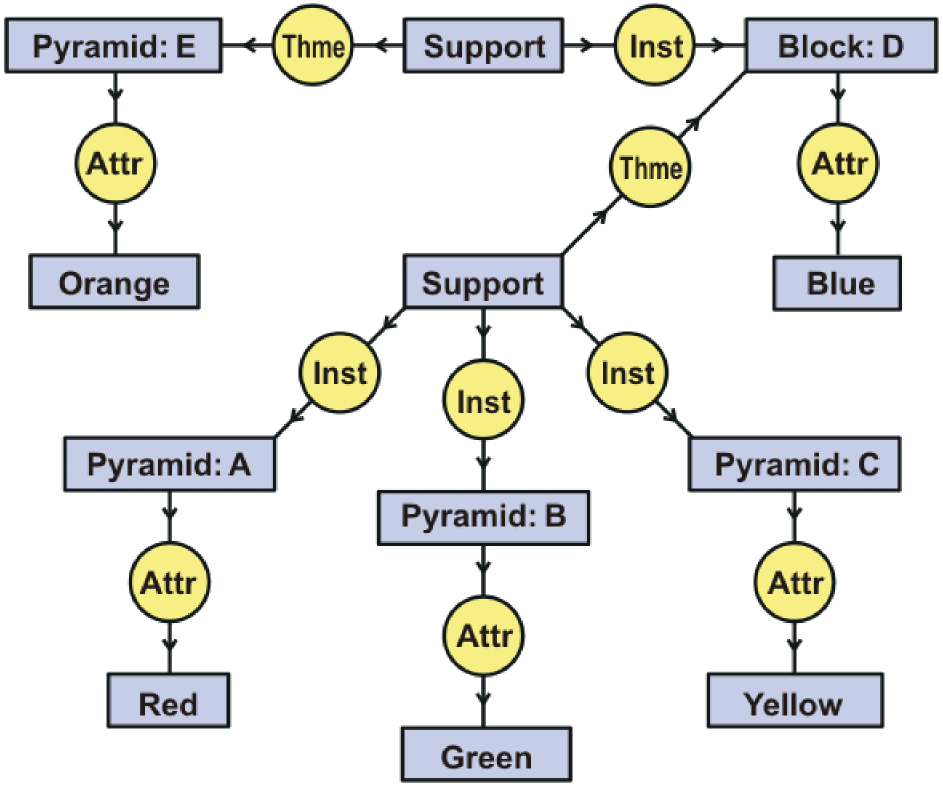
> “A red pyramid A, a green pyramid B, and a yellow pyramid C support a blue block D, which supports an orange pyramid E.”

The database is called structured, and English is called unstructured.

Yet English has even more structure, but of a very different kind.

Mapping English to a Conceptual Graph





“A red pyramid A, a green pyramid B, and a yellow pyramid C support a blue block D, which supports an orange pyramid E.”

The concepts (blue) are derived from English words, and the conceptual relations (yellow) from the case relations or thematic roles of linguistics.

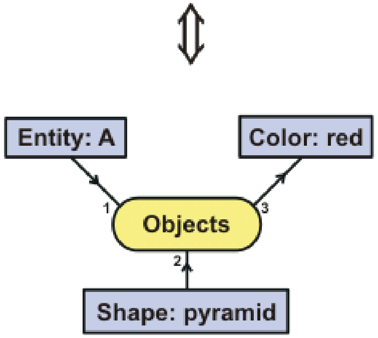
Mapping Database Relations to Conceptual Relations

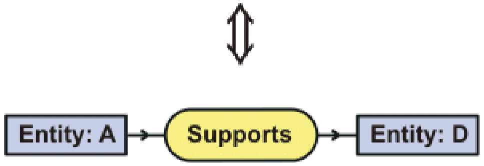
Objects

Entity	Shape	Color
A	pyramid	red
B	pyramid	green
C	pyramid	yellow
D	block	blue
E	pyramid	orange
F	block	blue
G	block	orange
H	block	blue

Supports

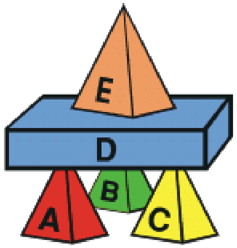
Supporter	Supportee
A	D
B	D
C	D
D	E
F	G
H	G

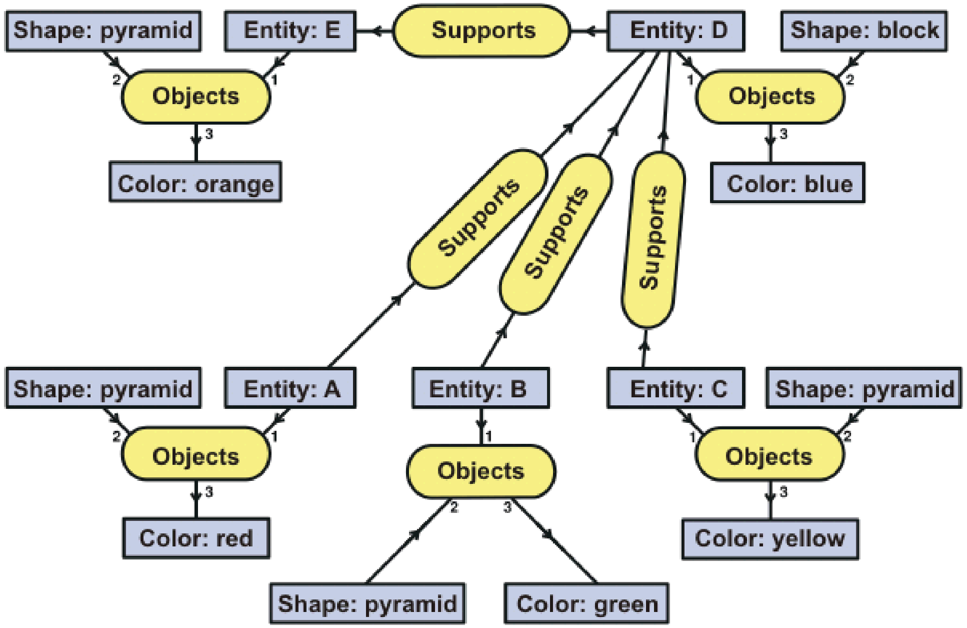




Each row of each table maps to one conceptual relation, which is linked to as many concepts as there are columns in the table.

Mapping an Entire Database to Conceptual Graphs





Shape: pyramid

Entity: E

Supports

Entity: D

Shape: block

Objects

Objects

Color: orange

Color: blue

Supports

Supports

Supports

Shape: pyramid

Entity: A

Entity: B

Entity: C

Shape: pyramid

Objects

Objects

Objects

Color: red

Color: yellow

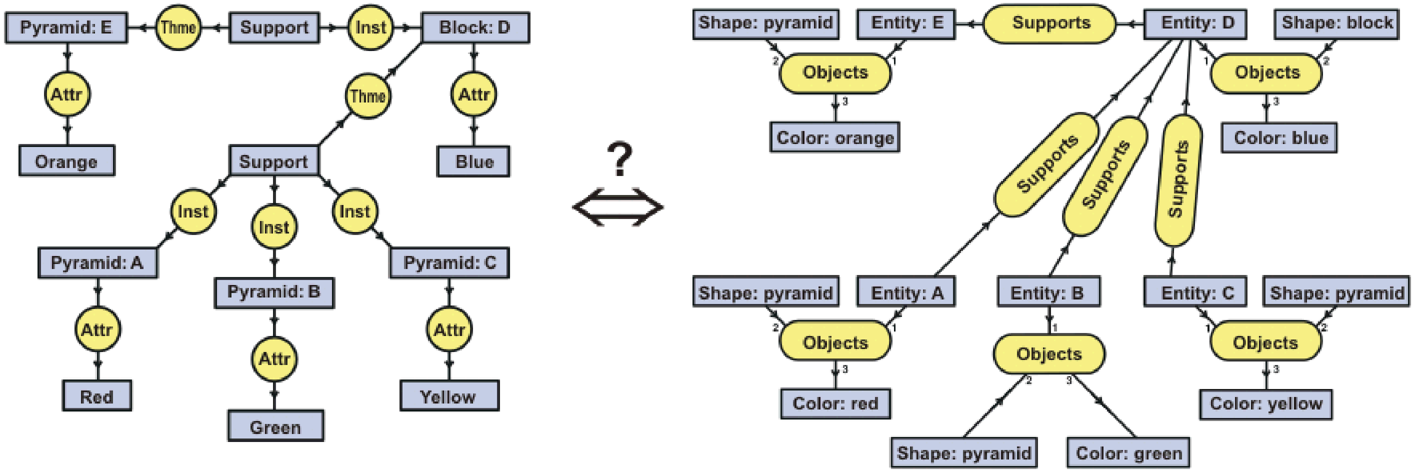
Shape: pyramid

Color: green

Join concept nodes that refer to the same entities.

Closely related entities are described by connected graphs.

Mapping the Two Graphs to One Another



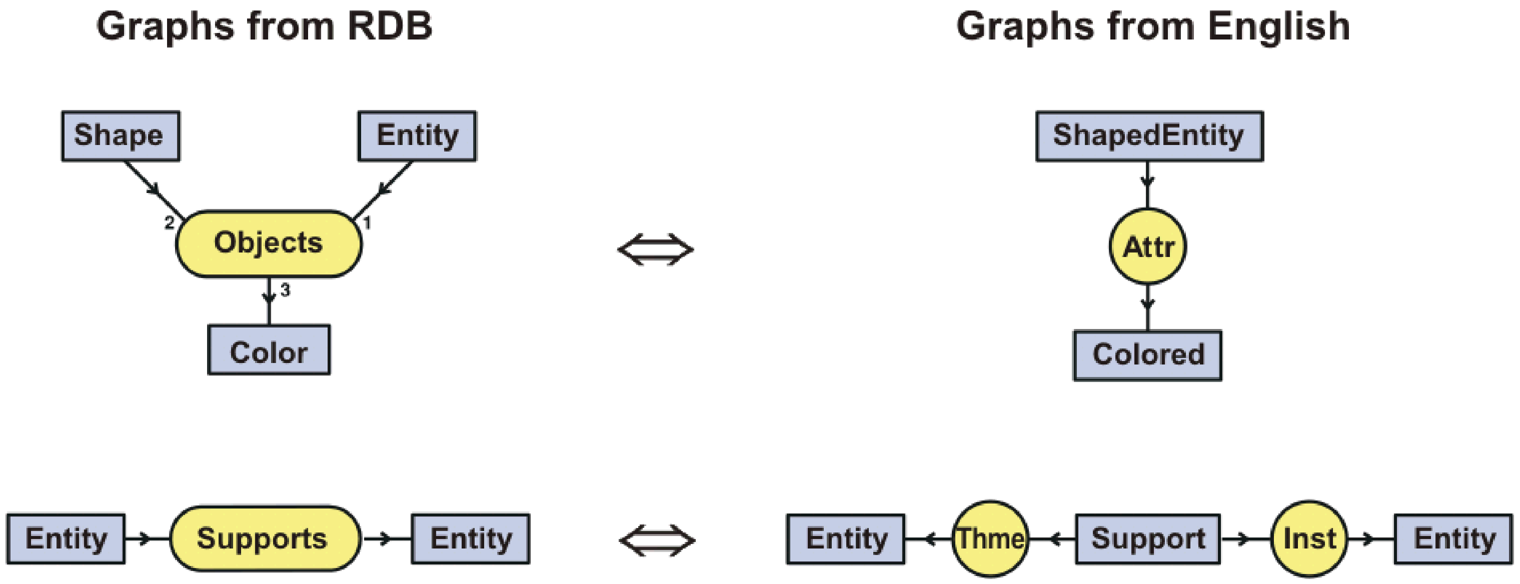
Very different ontologies: 12 concept nodes vs. 15 concept nodes, 11 relation nodes vs. 9 relation nodes, no similarity in type labels.

The only commonality is in the five names: A, B, C, D, E.

People can recognize the underlying similarities.

How is it possible for a computer to discover them?

Aligning Ontologies by Mapping Graphs



Graphs from RDB

Graphs from English

Repeated application of these two transformations completely map all nodes and arcs of each graph to the other.

This mapping, done by hand, is from an example by Sowa (2000), Ch 7.

The VivoMind Analogy Engine (VAE) found the mapping automatically.

See http://www.jfsowa.com/pubs/analog.htm

AbiWord Link Grammar Parser

A high-speed, broad coverage parser originally developed at CMU and also adapted for OpenOffice.

Generates graphs that are similar to conceptual graphs.

But its primary purpose is grammar checking:

- Designed for analyzing texts on any subject whatever.
- Uses little or no semantics.
- Links are labeled with syntactic tags, such as D, P, S, Sp, Ss...
- Has large numbers of special-purpose grammar rules.

The speed, coverage, and support by a broad community make it attractive.

But a considerable amount of computation is needed to translate the syntactic labels to semantic labels.

VivoMind Link Grammar

Syntactically much simpler than the AbiWord grammar:

- Replace the syntactic labels with names of conceptual relations, such as Agent, Patient, Theme, Instrument, Recipient...
- Delete the special-purpose grammar rules.
- The simpler grammar is more ambiguous, and it may leave many fragments of unattached CGs.
- But those fragments are linked together by matching graphs – either canonical graphs or other CGs derived from the documents.
- Graph matching can be supplemented with any knowledge sources that may be available.

Result: Sentences are mapped directly to conceptual graphs without an intermediate syntactic representation.

This “lightweight” grammar with “heavyweight” semantics can interpret many ungrammatical sentences.

See http://www.jfsowa.com/pubs/paradigm.pdf

Resolving Anaphoric References

An example that illustrates the task of linking references by proper names, pronouns, and noun phrases:

> ***Jon has a cat named Garfield and a dog named Odie. When they play games, the feline usually beats the canine and the human.***

Questions:

1. **Which individuals are included in the referent of 'they'?**
2. **What information is needed to link the noun phrases 'the feline', 'the canine', and 'the human' to their referents?**
3. **How is that information derived from the syntax and semantics?**

The VivoMind Language Processor maps all the sentences to conceptual graphs.

References are resolved by matching graphs and joining coreferent concept nodes.

Proto-Ontology Extractor

A tool for generating domain-dependent ontologies from one or more documents about any given subject:

1. Start with a fairly general ontology.
2. Use that ontology to translate one or more documents to CGs.
3. Analyze the results to find highly interconnected nodes.
4. Form a hypothesis that the subgraphs at those nodes represent concepts that are critical to the ontology of the subject.
5. Use the canonical formation rules to determine where to place those hypothetical concepts in a generalization hierarchy.
6. Generate a name for each concept based on the words from which the subgraph was derived.
7. Ask a subject matter expert to verify, modify, or reject the newly hypothesized concepts.
8. Repeat from step #2 with the new additions to the ontology.

Proto-Ontology Example

An excerpt from a document on biochemistry:

The movement of signals can be simple, like that associated with receptor molecules of the acetylcholine class: receptors that constitute channels which, upon ligand interaction, allow signals to be passed in the form of small ion movement, either into or out of the cell. These ion movements result in changes in the electrical potential of the cells that, in turn, propagates the signal along the cell. More complex signal transduction involves the coupling of ligand-receptor interactions to many intracellular events. These events include phosphorylations by tyrosine kinases and/or serine/threonine kinases.

Excerpts from the proto-ontology:

[intrinsic,enzymatic,activity]:sourceLineNumber(11)

[plasma,membrane]:sourceLineNumber(11)

[receptors,intrinsic,enzymatic]:sourceLineNumber(12)

[penetrate,plasma,membrane]:sourceLineNumber(11)

[affects,gene,transcription]:sourceLineNumber(19)

[ligand-receptor,complex,directly,affects]:sourceLineNumber(19)

Extending the Proto-Ontology

The proto-ontology consists of a collection of CGs derived from source documents and organized in a hierarchy.

Subject-matter experts can use Common Logic Controlled English (CLCE) to correct it or extend it:

- All computer output is in CLCE words, phrases, and sentences, supplemented with diagrams appropriate to the subject matter.
- All input from SMEs is either in menu selections or in CLCE words, phrases, constraints, rules, and questions.
- The SMEs never need to study logic, linguistics, or ontology.
- Their only training is one week of tutorials and guided practice working with documents of their own choosing.

The tools and methodologies are still under development, but SMEs can use the current versions effectively.

And they enjoy the experience.

Extracting Semantics, not Syntax

Statistical methods for language processing extract surface syntactic patterns from a training corpus:

- The syntactic patterns found in well-edited documents have little or no similarity to the patterns in emails or text messages.

But the proto-ontology extractor analyzes the underlying semantic patterns:

- The semantic patterns depend only on the subject matter.
- They are independent of the style or even the language.

For comparison, note the legacy re-engineering example in slides 24 to 27 of http://www.jfsowa.com/talks/pursue.pdf

The syntax of COBOL has very little similarity to English.

But semantic patterns extracted from COBOL programs were used to interpret English manuals and emails.

Automating Knowledge Acquisition

There are many ways to develop semantic systems, but no single best way for all applications.

Methodologies that require highly trained knowledge engineers are useful, but expensive.

Teaching SMEs how to use special ontology languages is often impractical and even a waste of their valuable time.

Newer tools enable SMEs to update and extend a knowledge base while using their native languages.

Technology under development will make such tools cheaper, easier to use, and more widely available.

Related Readings

Fads and Fallacies About Logic, by J. F. Sowa,
http://www.jfsowa.com/pubs/fflogic.pdf

Conceptual Graphs, by J. F. Sowa,
http://www.jfsowa.com/cg/cg_hbook.pdf

Two paradigms are better than one, but multiple paradigms are even better, by A. K. Majumdar & J. F. Sowa, http://www.jfsowa.com/pubs/paradigm.pdf

Pursuing the goal of language understanding, by A. K. Majumdar, J. F. Sowa, & J. Stewart, http://www.jfsowa.com/pubs/pursuing.pdf

Papers from a recent workshop on controlled natural languages,
http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-448/

Web site for controlled natural languages,
http://sites.google.com/site/controllednaturallanguage/

ISO/IEC standard 24707 for Common Logic,
http://standards.iso.org/ittf/PubliclyAvailableStandards/c039175_ISO_IEC_24707_2007(E).zip