Applying Robust Semantics

Ann Copestake

Computer Laboratory
University of Cambridge

September 2007

Outline.

Outline.

Applications: 1970s-1980s.

- Natural language interfaces to databases and knowledge bases:
  - Who had the highest sales figures in June 1982?
  - Is there a doctor on board the Vincennes?
- Exploit the limited domain: small lexicon, link to domain concepts, domain-specific ambiguity resolution. Database as denotation.

Applications: late 1980s onwards

- Information management:
  - Web search: return full documents (display snippets), generally little language processing.
  - Information Extraction (IE): relatively unrestricted text, specific types of information (e.g., company takeovers, terrorist incidents), instantiate fixed templates.
  - Question Answering (QA): general queries, match query to text/web.
- Broad-coverage, very shallow processing, mostly no compositional semantics.

Why use semantics in information management?

- Enables abstraction:
  - Paper 1: The synthesis of 2,8-dimethyl-6H,12H-5,11 methanodibenzo[b,f][1,5]diazocine (Troger's base) from p-toluidine and of two Troger's base analogs from other anilines
  - Paper 2: . . . Tröger's base (TB) . . . The TBs are usually prepared from para-substituted anilines
- Inference: e.g., search for papers describing Tröger's base syntheses which don't involve anilines?
- Domain and application independence.

Broad-coverage computational compositional semantics: present day.

- High-throughput parsers with some form of semantic output: CCG, RASP, ENJU, XLE . . . ERG/PET (medium throughput) . . .
- Effective statistical techniques for parse ranking (for syntactically different structures).
- Robust entailment as a common basis for applications.
- Links to ontologies/semantic web.
- More ‘stuff’ online, increased need for precision.

What is Compositional Semantics?

Topics include:

- Predicate-argument structure (nouns, adjectives as well as verbs).
  Scopal (e.g., *probably*) vs non-scopal (e.g., *quickly*).
- Construction semantics: relative clauses, appositives, tag questions, pseudo-partitives . . .
- Tense, aspect, distributivity, generics vs individual reference, mass/count.
- Non-compositional multi-word expressions.
- Maybe: derivational morphology, sense extension.

Not: meaning of open-class words.

Compositional Semantics: working definition

Meaning information that can be associated with syntax and morphology.

- Fully identified (for English): Predicate-argument structure, modifier scope, some constructions.
- Partially identified: quantifier scope, compound nouns, tense, aspect, massness, genericity, sense extension.

Partial information, e.g. genericity:

Brontosaurus ate half a ton of vegetation a day
the Brontosaurus ate a sailor, but it was a herbivore
your brontosaurus ate my palm tree

Implications of broad-coverage processing for computational semantics.

- Semantic processing is relatively shallow. No underlying knowledge base for disambiguation.
- Detailed lexical information is not available. At best, irregular morphology, syntactic subcategorization for frequent word senses, WordNet and/or FrameNet. Incomplete/absent: multiword expressions, mass terms, verb aspect, pseudo-partitive constructions ...
- Support inter-sentential anaphora/text structure.
- Avoid semantics multiplying readings: underspecification.

Underspecification and Sudoku solving

			7					8
		9					2	
	5			3			9	
8					2			
		6				7		
			4					1
	3			9			6	
	2					4		
7					1			

Solving.

			7					8
		9					2	
	5			3			9	
8					2			
		6				7		
			4					1
	3			9			6	
	2					4		
7					1			

Possibility 1.

			7					8
		9					2	7
	5			3			9	
8					2			
		6				7		
			4					1
	3			9			6	
	2					4		
7					1			

Possibility 2.

			7					8
		9					2	
	5			3			9	7
8					2			
		6				7		
			4					1
	3			9			6	
	2					4		
7					1			

Underspecification.

			7					8
		9					2	7
	5			3			9	7
8					2			
		6				7		
			4					1
	3			9			6	
	2					4		
7					1			

Inference on underspecified form.

			7					8
		9					2	7
	5			3			9	7
8					2			
		6				7		
			4					1
	3			9			6	
	2					4		
7					1			

Inference on underspecified form.

			7					8
		9					2	7
	5			3			9	7
8					2			
		6				7		
			4					1
	3			9			6	
	2					4	7	
7					1			

Some examples of underspecification in computational semantics.

- Quantifier scope: single underspecified reading from each syntactic analysis.
- Genericity, massness, aspect.
- Compound nominal relations: general relationship.
- Prepositional phrase attachment: limit syntactic ambiguity.
- Word senses: hierarchy of word senses.
- Feature values: hierarchy of values. Underspecification for morphology vs semantically coherent classes.

Inference and robust semantics

- Inference motivates use of semantic representations.
- BUT:
  - Inference on underspecified representations?
  - Higher-order constructs?
  - Limited speed of theorem provers.
  - No closed world assumption (in contrast to database query).
  - Not robust to missing information.
- SO: pattern matching operations on semantics . . .

Inference and robust semantics

- Inference motivates use of semantic representations.
- BUT:
  - Inference on underspecified representations?
  - Higher-order constructs?
  - Limited speed of theorem provers.
  - No closed world assumption (in contrast to database query).
  - Not robust to missing information.
- SO: pattern matching operations on semantics ...

Applying robust semantics in DELPH-IN

Related work: PARC/Powerset, Moldovan et al, Bos et al. etc
This talk:

- MRS/RMRS approach to semantic representation.
- Abstract operations.
- Various applications.
- Relationship to ‘proper’ inference.

Semantic operations on (R)MRS have evolved and expanded: emphasis on practical utility, not theory.

Outline.

DELPH-IN: Deep Linguistic Processing using HPSG

- Informal collaboration on tools and grammars: see `http://www.delph-in.net/`
- Large grammars for English, German and Japanese; medium/growing for Spanish, Norwegian, Portuguese, Korean, French. Many small grammars.
- Common semantic framework: Minimal Recursion Semantics (MRS) and Robust MRS. RMRS also from shallower parsing, chunking, POS tagging.
- Parsing and generation (realization), integrated shallower processing.
- Grammar Matrix: framework/starter kit for the development of grammars for diverse languages.

Some recent projects using MRS/RMRS

- DeepThought: Information Extraction, email response
- LOGON: Norwegian-English MT (semantic transfer)
- SciBorg: IE from Chemistry texts
- Reasoning about meetings (Schlangen et al, 2003)
- Dridan (2006), Dridan and Bond (2006): Question Answering (also Watson et al (2003))
- QUETAL: QA from structured knowledge (Frank et al)
- Herbelot and Copestake (2006): Ontology extraction from Wikipedia
- Nichols, Bond, Flickinger (2005): Ontology extraction from MRDs

Semantic representation: MRS

The mixture was allowed to warm to room temperature.

⟨ l3:_the_q(x5,h6,h4), l7:_mixture_n(x5),
l9:_allow_v_1(e2,u11,x5,h10), l13:_warm_v_1(e14,x5),
l13:_to_p(e15,e14,x16), l17:udef_q(x16,h18,h19),
l20:compound(e22,x16,x21), l23:udef_q(x21,h24,h25),
l26:_room_n(x21), l20:_temperature_n(x16) ⟩
⟨ qeq(h6,l7), qeq(h18,l20), qeq(h24,l26), qeq(h10,l13) ⟩

MRS: main features

- Flat: list of EPs (each with label), list of qeqs.
- Underspecified quantifier scope: labels and holes, linked with qeqs (equality modulo quantifiers).
  l9:_allow_v_1(e2,u11,x5,h10), qeq(h10,l13),
  l13:_warm_v_1(e14,x5)
- Conjunction from modification etc indicated by shared labels: l13:_warm_v_1(e14,x5), l13:_to_p(e15,e14,x16)
- Lexical predicates (leading underscore): lexeme, coarse sense (POS), fine sense.
- Construction predicates (e.g., compound).
- Sorted variables: tense, etc (and simple information structure).

One of the scoped forms

The mixture was allowed to warm to room temperature.

_the_q(x5,_mixture_n(x5),
 _allow_v_1(e2,u11,x5,
  udef(x21, _room_n((x21),
   udef(x16,compound(e22,x16,x21) ∧ _temperature_n(x16),
    _warm_v_1(e14,x5) ∧ _to_p(e15,e14,x16)))))

Semantic representation: RMRS

The mixture was allowed to warm to room temperature.

⟨ l3:a1:_the_q(x5), l7:a2:_mixture_n(x5), l9:a3:_allow_v_1(e2), l13:a5:_warm_v_1(e14), l13:a6:_to_p(e15), l17:a7:udef_q(x16), l20:a8:compound(e22), l23:a9:udef_q(x21), l26:a10:_room_n(x21), l20:a11:_temperature_n(x16)⟩

⟨ a1:RSTR(h6), a1:BODY(h4), a3:ARG2(x5), a3:ARG3(h10), a5:ARG1(x5), a6:ARG1(e14), a6:ARG2(x16), a7:RSTR(h18), a7:BODY(h19), a8:ARG1(x16), a8:ARG2(x21), a9:RSTR(h24), a9:BODY(h25) ⟩

⟨ qeq(h6,l7), qeq(h18,l20), qeq(h24,l26), qeq(h10,l13) ⟩

MRS vs RMRS

- l9:_allow_v_1(e2,u11,x5,h10) in MRS
  l9:a3:_allow_v_1(e2), a3:ARG2(x5), a3:ARG3(h10) in RMRS.
- Further factorization: separation of arguments.
- All EPs have an anchor which relates args to EPs.
- RMRS can omit or underspecify ARGs: robust to missing lexical information.

Character positions

The mixture was allowed to warm to room temperature.

⟨ l3:a1:_the_q(x5)$_{\langle 0,3\rangle}$, l7:a2:_mixture_n(x5)$_{\langle 4,11\rangle}$,
l9:a3:_allow_v_1(e2)$_{\langle 16,23\rangle}$, l13:a5:_warm_v_1(e14)$_{\langle 27,31\rangle}$,
l13:a6:_to_p(e15)$_{\langle 32,34\rangle}$, l17:a7:udef_q(x16)$_{\langle 35,52\rangle}$,
l20:a8:compound(e22)$_{\langle 35,52\rangle}$, l23:a9:udef_q(x21)$_{\langle 35,52\rangle}$,
l26:a10:_room_n(x21)$_{\langle 35,39\rangle}$, l20:a11:_temperature_n(x16)$_{\langle 40,52\rangle}$⟩
⟨ a1:RSTR(h6), a1:BODY(h4), a3:ARG2(x5), a3:ARG3(h10),
a5:ARG1(x5), a6:ARG1(e14), a6:ARG2(x16), a7:RSTR(h18),
a7:BODY(h19), a8:ARG1(x16), a8:ARG2(x21), a9:RSTR(h24),
a9:BODY(h25) ⟩
⟨ qeq(h6,l7), qeq(h18,l20), qeq(h24,l26), qeq(h10,l13) ⟩

RMRS from POS tagger

The mixture was allowed to warm to room temperature.

⟨ l1:a2:_the_q(x3), l4:a5:_mixture_n(x6), l7:a8:_allow_v(e9), l10:a11:_warm_v(e12), l13:a14:_to_p(e15), l16:a17:_room_n(x18), l19:a20:_temperature_n(x21)⟩

⟨⟩

⟨⟩

All variables distinct, no ARGs, no qeqs.

Chunker: equate nominal indices, etc.

RMRS as semantic annotation of lexeme sequence.

- Annotate most lexemes with random label, anchor, arg0. Note: null semantics for some words, e.g., infinitival *to*.
- Partially disambiguate lexeme with n, v, q, p etc.
- Add sortal information to arg0.
- Implicit conjunction: add equalities between labels.
- Ordinary arguments: add ARGs (possibly underspecified) between anchors and arg0.
- Scopal arguments: add ARG plus qeq between anchors and labels.

Standoff annotation on original text via character positions.

RMRS Elementary Predication

An RMRS EP contains:

1. the label of the EP: this is shared by other EPs to indicate implicit conjunction.
2. an anchor, not shared by any other EPs.
3. a relation
4. up to one argument of the relation (the arg0)

This is written as label:anchor:relation(arg0).
l13:a5:_warm_v_1(e14)
l13:a6:_to_p(e15)

RMRS ARGs

An RMRS ARG relation contains:

1. an anchor, which must also be the anchor of an EP.
2. an ARG relation, taken from a fixed set (here: ARG1, ARG2, ARG3, RSTR, BODY, plus the underspecified relations: ARG1-2, ARG1-3, ARG1-2, ARG2-3, ARGN).
3. exactly one argument. This must be 'grounded' by an EP: i.e., if it is a normal variable it must be the ARG0 of an EP, or if it is a hole, it must be related to the label of an EP by a qeq constraint.

a5:ARG1(x5), l13:a5:_warm_v_1(e14), l7:a2:_mixture_n(x5)

RMRS structures

An RMRS structure contains:

1. rels: The bag of EPs.
2. args: The bag of argument relations.
3. hcons: qeq constraints. A qeq relationship always holds between a hole in an argument relation and the label of an EP.

Outline.

RMRS Matching

lb1:every_q(x),
lb1:RSTR(h9),
lb1:BODY(h6),
lb2:cat_n(x),
lb4:some_q(y),
lb1:RSTR(h8),
lb1:BODY(h7),
lb5:dog_n_1(y),
lb3:chase_v(e),
lb3:ARG1(x),
lb3:ARG2(y)

lb1:every_q(x),
lb1:RSTR(h9),
lb1:BODY(h6),
lb2:cat_n(x),
lb4:some_q(y),
lb1:RSTR(h8),
lb1:BODY(h7),
lb5:dog_n_1(y),
lb3:chase_v(e),
lb3:ARG1(x),
lb3:ARG2(y)

RMRS Matching

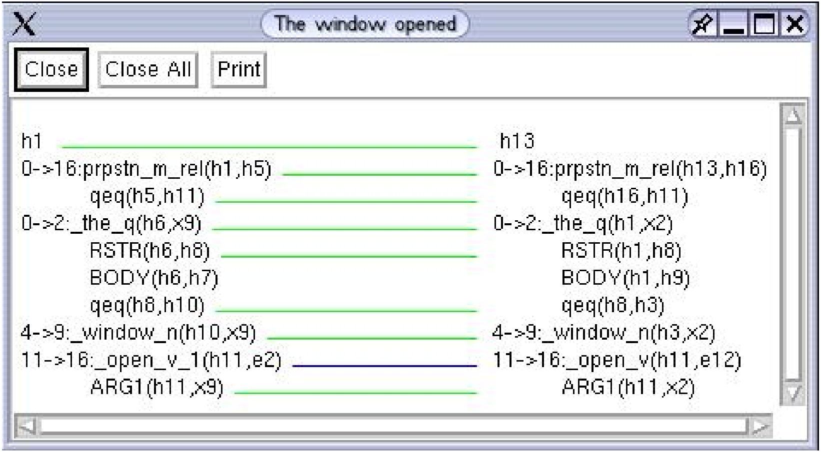
lb1:every_q(x),
lb1:RSTR(h9),
lb1:BODY(h6),
lb2:cat_n(x),
lb4:some_q(y),
lb1:RSTR(h8),
lb1:BODY(h7),
lb5:dog_n_1(y),
lb3:chase_v(e),
lb3:ARG1(x),
lb3:ARG2(y)

lb1:every_q(x),
lb1:RSTR(h9),
lb1:BODY(h6),
lb2:cat_n(x),
lb4:some_q(y),
lb1:RSTR(h8),
lb1:BODY(h7),
lb5:dog_n(y),
lb3:chase_v(e),
lb3:ARG1-2(x),

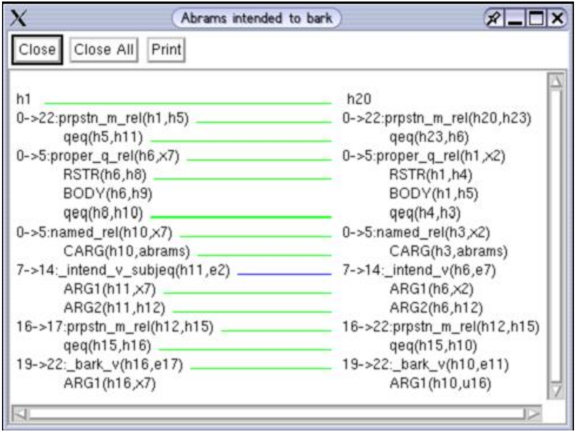
RMRS Matching

lb1:every_q(x),	lb1:every_q(x),
lb1:RSTR(h9),	
lb1:BODY(h6),	
lb2:cat_n(x),	lb2:cat_n(x),
lb4:some_q(y),	lb4:some_q(y),
lb1:RSTR(h8),	
lb1:BODY(h7),	
lb5:dog_n_1(y),	lb5:dog_n(y),
lb3:chase_v(e),	lb3:chase_v(e)
lb3:ARG1(x),	
lb3:ARG2(y)	

ERG-RASP comparison 1



ERG-RASP comparison 2



RMRS EP matching

An EP1 matches EP2 if:

1. the relation associated with EP1 is compatible with the relation associated with EP2. ‘compatibility’: partial order on relations.
2. the arg0 associated with EP1 is compatible with the arg0 associated with EP2 (including sortal properties)
3. Neither of the anchors are already matched.

If EP1 matches EP2, variable equivalences are:
$l_1/l_2, a_1/a_2, arg0_1/arg0_2$.
Full set of variable equivalences from matching two RMRSs: RMRS1/RMRS2.

RMRS ARG matching

Matching argument relations depends on RMRS1/RMRS2.
ARG-REL1 matches ARG-REL2 iff:

1. the anchor of ARG-REL1 is bound to the anchor of ARG-REL2 in RMRS1/RMRS2
2. and, if the argument of ARG-REL1 is a normal variable, it is bound to the argument of ARG-REL2 in RMRS1/RMRS2
3. or, if the argument of ARG-REL1 is a hole, it is qeq a label which is bound to a label l2 in RMRS1/RMRS2 such that the argument of ARG-REL2 is qeq l2
4. and the relation in ARG-REL1 is compatible with the relation in ARG-REL2

RMRS matching

RMRSs R1 and R2 match iff:

- each EP in R1 matches an EP in R2
- each EP in R2 matches an EP in R1
- each argument relation in R1 matches an argument relation in R2
- each argument relation in R2 matches an argument relation in R1

(BODY arguments are generally ignored: unscoped representations)

RMRS matching variants

- RMRSs may be checked for subsumption rather than compatibility: e.g., idiom patterns.
- RMRS patterns may be used rather than matching two RMRSs derived by a grammar: e.g., information extraction.
- Unmatched EPs allowed on one or both sides: e.g., QA.
- Robust weighted match: score according to which EPs match and whether their arguments match: e.g., QA.
- A match may signal some action: e.g., 'null semantic items' for generation.

Matching is a component of merging and 'munging'.

RMRS merging

- Two matching RMRSs may be merged: conjunction.
- Merging for patching up a partial deep analysis: Heart of Gold (Ulrich Schäfer).
- Packing partially compatible RMRSs into a lattice.
- Merging uniqueness for parse results guaranteed by ordering of EPs in analysis.

(R)MRS ‘munging’

- Rules for mapping between (R)MRSs
- Originally a hack for Verbmobil, later found many uses . . .
- Rules: input, output, context (all optional).
  If the input matches part of an MRS, and the context also matches, then the input is converted to the output.
- Each rule applied multiple times to one MRS, rules applied in a sequence in a ruleset, no reapplication of rulesets.
- Refined by Stephan Oepen for LOGON semantic transfer: also monolingual paraphrase, mapping input to a domain-specific representation (e.g., Schlangen et al).

Outline.

Generation / realization

Generate all and only the strings with ‘compatible’ semantics:

- If LF1 is generated by grammar G from string S, and LF2 is logically equivalent to LF1, then a realiser working with grammar G should accept LF2 and produce string S
- Unfortunately impossible for even first order predicate calculus (pointed out by Shieber)
- RMRS matching criterion instead: output has same predications, equivalence of ‘grammatical’ conjunction. Broaden this by underspecification.

Lexicalist generation (simplified!)

[a(y), consultant(y), german(y), every(x), manager(x), interview(e,x,y)]

1. For each elementary predication, find a corresponding lexical entry.
2. Set the argument positions in the lexical entry. to constant values - e.g., interview(c2,c1,c3), manager(c1), consultant(c3), german(c3), a(c3), every(c1)
   This means that unification ensures that the predicate-argument structure is correct.
3. Generate by parsing different orders of lexical items.

Generation chart (simplified!)

1. no overlap: check as the edges are constructed that EPs are only used once
2. completeness check at the end
3. some restrictions on the grammar:
   - daughters may not overlap — e.g., cannot have semantics constructed by means of multiple inheritance between types contributed from two sources
   - monotonicity: none of the components may be removed when constructing a phrase

Null semantics in generation

- Some lexical entries (e.g., infinitival *to*, expletive *there*) have no associated EP
- introduce on the basis of null semantics rules triggered by match on the input MRS
- l:a:_be_v_there_rel(e) −> `"there_expl"` where `"there_expl"` is a lexical entry identifier.

Idioms

- Most idioms are ‘compositional’: meaning of the idiomatic phrase can be treated as composed of the meaning of the component parts, with weird senses.
- *take heart*, *spill beans*, *cat out of (the) bag*. e.g., *spill the beans* corresponds roughly to *reveal the secrets*.
- Syntactic variation:

  We take considerable heart from the knowledge . . . (from BNC)

- Idiomatic senses as normal lexical entries with use constrained by semantic patterns acting as root conditions: l:a:_take_v_i(e), ARG2(a,x), l1:a1:_heart_n_i(x)

Outline.

Questions and answers: QA, NLID etc

A valid answer should entail the query (with suitable interpretation of *wh*-terms etc).
Is a dog barking?
$\exists x[\text{dog}'(x) \wedge \text{bark}'(x)]$

A dog is barking entails *A dog is barking*

Rover is barking and *Rover is a dog* entails *A dog is barking.*
$\text{bark}'(\mathit{Rover}) \wedge \text{dog}'(\mathit{Rover})$ entails $\exists x[\text{dog}'(x) \wedge \text{bark}'(x)]$

which dog is barking?
$\text{bark}'(\mathit{Rover}) \wedge \text{dog}'(\mathit{Rover})$ entails $\exists x[\text{dog}'(x) \wedge \text{bark}'(x)]$
Bind query term to answer.

QA example 1

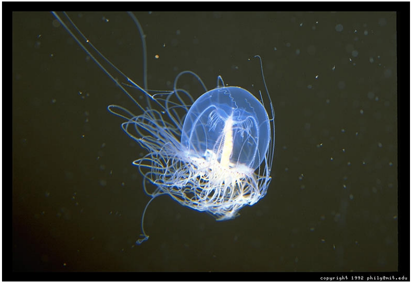
Example

What eats jellyfish?

QA example 1

Example

What eats jellyfish?



QA example 1

Example

What eats jellyfish?

Pattern matching on RMRS:
[a:eat(e), ARG1(a,x), ARG2(a,y), jellyfish(y)]
So won't match on *jellyfish eat fish*.

What eats jellyfish?

Example

Turtles eat jellyfish and they have special hooks in their throats to help them swallow these slimy animals.

What eats jellyfish?

Example

Turtles eat jellyfish and they have special hooks in their throats to help them swallow these slimy animals.

Match on [a:eat(e), ARG1(a,x), ARG2(a,y), jellyfish(y)]

A logically valid answer which entails the query since the conjunct can be ignored.

What eats jellyfish?

Example

Turtles eat jellyfish and they have special hooks in their throats to help them swallow these slimy animals.

Turtles again

What eats jellyfish?

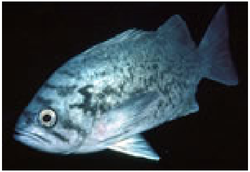
Example

Sea turtles, ocean sunfish (Mola mola) and blue rockfish all are able to eat large jellyfish, seemingly without being affected by the nematocysts.

What eats jellyfish?

Example

Sea turtles, ocean sunfish (Mola mola) and blue rockfish all are able to eat large jellyfish, seemingly without being affected by the nematocysts.



What eats jellyfish?

Example

Sea turtles, ocean sunfish (Mola mola) and blue rockfish all are able to eat large jellyfish, seemingly without being affected by the nematocysts.

Pattern matching on RMRS:
[a:eat(e), ARG1(a,x), ARG2(a,y), large(y), jellyfish(y)]

eat large jellyfish entails *eat jellyfish* (because *large* is intersective)

What eats jellyfish?

Example

Also, open ocean-dwelling snails called Janthina and even some seabirds have been known to eat jellyfish.

What eats jellyfish?

Example

Also, open ocean-dwelling snails called Janthina and even some seabirds have been known to eat jellyfish.

What eats jellyfish?

Example

Also, open ocean-dwelling snails called Janthina and even some seabirds have been known to eat jellyfish.

[a1:know(e), ARG2(a1,h1), qeq(h1,lb), lb:a:eat(e), ARG1(a,x), ARG2(a,y), jellyfish(y)]

Logically valid if *know* is taken as truth preserving.

$$\forall P \forall y [know(y, P) \implies P]$$

Axioms like this required for logically valid entailment: missing axiom would cause failure to match.

QA Example 2

Example

What is the largest town in Cornwall?

Interface to database of Cornish towns could use numerical population values and calculate this.

QA: assumption is the data is directly available in some text (no closed world assumption)

QA Example 2

Example

What is the largest town in Cornwall?

Interface to database of Cornish towns could use numerical population values and calculate this.
QA: assumption is the data is directly available in some text (no closed world assumption)

What is the largest town in Cornwall?

St Austell is Cornwall's largest town and a centre of the china clay industry.

Query: [named(x,"Cornwall"), in(e,x,y), large(e,y), superl(e1,e), town(y)]
Answer: [named(x,"Cornwall"), poss(e,y,x), large(e,y), superl(e1,e), town(y)]
So strict match misses here where word match would work.
Actual QA experiments: weighted match. Closed-class words and construction relations are given less weight than matches on EPs derived from lexemes.
Better alternative long term: set of valid equivalence rules (poss as underspecified relation).

What is the largest town in Cornwall?

> *In spite of its city statue (sic), Truro is not the largest town in Cornwall; there are several larger agglomerations.*

Negation: like *know* in earlier example, but here simple pattern matching gets it wrong.
Contexts which block match (versus axioms which allow entailment for theorem proving).

What is the largest town in Cornwall?

Penzance is the largest town in west Cornwall.

west can be treated as intersective, but this does not imply

Penzance is the largest town in Cornwall.

Superlatives require a notion of the comparison set (not in current ERG/RMRS representation).

What is the largest town in Cornwall?

Penzance is the largest town in west Cornwall.

west can be treated as intersective, but this does not imply

Penzance is the largest town in Cornwall.

Superlatives require a notion of the comparison set (not in current ERG/RMRS representation).

What is the largest town in Cornwall?

St Austell is the largest town in Cornwall, in terms of population.

Dimensionality of adjectives: this is hard!

What is the largest town in Cornwall?

> *Also at [URL], St Austell is said to be the largest town, however the population figure is doubtful.* (from Wikipedia Portal talk)

Contexts which might block an inference may be indefinitely far away in a document, maybe even in a different document.
Cf scientific texts: want to extract citations which contradict a paper. Trustworthiness of documents: implausible can be obtained by detailed textual analysis with current technology, but approximations.

What is the largest town in Cornwall?

> *Also at [URL], St Austell is said to be the largest town, however the population figure is doubtful.* (from Wikipedia Portal talk)

Contexts which might block an inference may be indefinitely far away in a document, maybe even in a different document. Cf scientific texts: want to extract citations which contradict a paper. Trustworthiness of documents: implausible can be obtained by detailed textual analysis with current technology, but approximations.

Matching in QA: Summary

- Word overlap: no account of the context of the query words in the answer.
- Simple RMRS matching: context relating the query words, but no context from the remainder of the sentence.
- Refined RMRS matching: check for specific types of context, such as negation.
- Full sentence-based entailment takes into account the sentence context, but not document context or inter-document context.

RMRS: augment language with explicit semantics to different extents.
Robust matching: augment bag-of-words technique to different extents.

Outline.

The Semantic Web

- Like NL search, QA etc, semantic web querying:
  - cannot rely on a closed world assumption
  - requires mapping between representations
- Claim: language processing will soon just be needed for old texts. All new publication will use semantic markup.
- But: agreement on semantic markup languages is limited. Even scientific publishing is not simply about facts.
- 'Information Layer' (Spärck Jones 2007): connection via words may be good enough for many tasks.
- Semantic web markup best seen as an addition to natural language, not a replacement.
  Computational semantics: enrich texts to make aspects of meaning more accessible to subsequent processing.

Concluding comments

- Computational semantic representations can be robust to missing information, especially missing lexicon.
- Flat, ‘surfacy’ representations: more robustness, easier processing, semantics as annotation of natural language rather than replacement for it.
- Semantics is useful in applications even without ‘proper’ inference. Semantic operations can be robust to missing ‘axioms’.
- Going deeper:
  - Lexical semantics: symbolic relationships between predicates, vector space model of predicates.
  - Discourse relations, anaphora, context.

Credits

Dan Flickinger, Simone Teufel, CJ Rupp, Ben Waldron, Advaith Siddharthan, Ted Briscoe, John Carroll, Ivan Sag, Carl Pollard, Anette Frank, Alex Lascarides, David Schlangen, Stephan Oepen, Emily Bender, Rob Malouf, Francis Bond, Tim Baldwin, Aline Villavicencio, Melanie Siegel, Lars Hellan, Dorothee Beerman, Ulrich Callmeier, Ulrich Schäfer, Bernd Kiefer, Victor Poznanski, Susanne Riehemann, Anna Ritchie, Rebecca Dridan, Aurelie Herbelot, Richard Bergmair

with funding from BMBF, CSLI IAP, NSF, EPSRC, NTT, European Commission and Boeing.