Revisiting Natural Language Interfaces to Databases

Michael Minock
Department of Computing Science,
University of Umeå
E-mail: mjm@cs.umu.se
Introduction

- The basic problem:
  - configuration:
    DB + linguistic knowledge → NL Interface
  - querying:
    NL queries + NL Interface → NL answers

- Some extra requirements:
  - query paraphrasing
  - update operations
  - configuration tools
A Great Problem for Computational Linguistics

- Semantics restricted to a well behaved case:
  - may evaluate the performance of various syntactic theories
  - controlled environment for study of anaphora, ellipsis, dialogue, context...
- Could be useful
  NL could give advantage over forms based, metaphor based or formal language interfaces
  - people already know natural language - no training!
  - forms and metaphor based approaches have difficulty with negation, quantification and high conceptual complexity
  - meta-level questions
  - dialogue and query refinement
- DEMO! www.cs.umu.se/~mjm/step

\(^a\)Key word search is generally too weak for relational databases.
Overview

• Basic notions:
  – databases
  – semantic interpretation

• Basic approaches:
  0. syntax based systems
  1. semantic grammars
  2. transportable systems

• Some history... what happened?

• Recent work:
  – Microsoft English Query
  – Precise
  – STEP

• Evaluation

• Future prospects and summary
Question: What is a ‘database’?

Answer: “A relational database!”

A set of relations \( \mathbf{R} = R_1, \ldots, R_n \), each with an associated arity

A database state \( D \) is the set of extensions (i.e. tuple sets) for each relation

\( \Sigma \) is the set of functional dependency (primary key) and inclusion dependency (foreign key) constraints

\( \Sigma \models D \) when \( D \) does not violate \( \Sigma \)

A query is \( Q \) a formula over some set of \( m \) free variables

Answers to \( Q \), \( A = Q(D) \) are \( m \)-tuples that satisfy the query over the structure associated with \( D \)
An Example Database

Schema:
Restaurant(id, name, address, city, phone)
Type(id, type)
Location(id, lat, long)

Queries:
“List the Italian restaurants”:
{x|Restaurant(x)\(\land\) 
  \((\exists y)(Type(y) \land x.id = y.id \land y.type = 'Italian')\)}

“Give the phone number of Rex?”:
{x.phone|Restaurant(x) \land x.name = 'Rex'}
What is Semantic Interpretation?

\[ I(W, C) \rightarrow \{ Q_1, ..., Q_m \} \]

where \( W = w_1, ... w_n \) is a sequence of words, \( C \) is the context\(^a\) and \( Q_1, ..., Q_m \) are logically distinct queries.

For a given population of sentences \( \Omega \):

\( I \) is ‘sound’ if for all \( W \in \Omega \) and all \( Q \in I(W, C) \), \( Q \) is a reasonable interpretation of \( W \) in the context \( C \).

\( I \) is ‘complete’ if for all \( W \in \Omega \), for all \( Q \) that are reasonable interpretations of \( W \) in the context \( C \), \( Q \in I(W, C) \).

Accuracy of \( I \) for a representative sample of \( \Omega' \subseteq \Omega \) is ...

\(^{a}\)We shall only consider \( C = \emptyset \) here.
Syntax Based Systems

- Parse word sequence to parse tree(s)
- Use ‘knowledge’ to map from parse tree(s) to database queries.

\[ S \rightarrow VP \]
\[ VP \rightarrow V \cdot NP \]
\[ NP \rightarrow ProperNoun|Det \cdot Nominal \]
\[ Nominal \rightarrow Nominal \cdot PP|N|ADJ \cdot Nominal \]
\[ PP \rightarrow P \cdot NP \]
\[ Det \rightarrow \text{“the”} \]
\[ P \rightarrow \text{“of”} \]
\[ V \rightarrow \text{“list”|“give”} \]
\[ N \rightarrow \text{“phone number”|“restaurants”} \]
\[ ProperNoun \rightarrow \text{“Rex”} \]
\[ ADJ \rightarrow \text{“Italian”} \]

- Mapping knowledge is notoriously difficult to encode
- High degree of ambiguity in large grammars
- Not bi-directional
Semantic Grammars

- Basing ‘syntactic categories’ on domain concepts

$S \rightarrow AR \quad \{\text{Query}(AR.sem)\}$
$S \rightarrow RS \quad \{\text{Query}(RS.sem)\}$
$AR \rightarrow \text{“give the”} PS \text{“of”} R \text{“?”} \quad \{PS.sem | R.sem\}$
$PS \rightarrow \text{“phone number”} \quad \{x.phone\}$
$PS \rightarrow \text{“address”} \quad \{x.address\}$
$RS \rightarrow \text{“list the ”} R \quad \{x | R.sem\}$
$R \rightarrow $C_1$\text{“restaurants”}$
   \{Restaurant(x) \land$
   $ (\exists y)(Type(y) \land x.id = y.id \land y.type = $C_1$)\}$
$R \rightarrow \text{“restaurants”} \quad \{Restaurant(x)\}$
$R \rightarrow $C_1 \quad \{Restaurant(x) \land x.name = $C_1$\}$

+ Can give surprisingly robust solutions in closed domains

- Must build grammar for each new schema

- A black art

- Not (immediately) bi-directional
Transportable Systems

- Uses very large domain independent grammars (with attached semantics).
- Maps to some domain independent representation language.
- *Translation knowledge* maps the abstract meaning representations to queries over the database relations.

(adapted form Copskate, et al. 1990)
Example A Domain Independent Grammar

\[ S \rightarrow VP \quad \{ \text{IMP}(VP.sem(\text{Hearer})) \} \]
\[ VP \rightarrow V \cdot NP \quad \{ V.sem(NP.sem) \} \]
\[ NP \rightarrow \text{ProperNoun} \quad \{ \text{ProperNoun.sem} \} \]
\[ NP \rightarrow \text{Det} \cdot \text{Nominal} \quad \{ \langle \text{Det.semxNominal.sem} \rangle \} \]
\[ \text{Nominal} \rightarrow \text{Nominal} \cdot \text{PP} \]
\[ \quad \{ \lambda z.\text{Nominal.sem}(z) \land \text{PP.sem}(z) \} \]
\[ \text{Nominal} \rightarrow \text{Noun} \quad \{ \lambda x.\text{ISA}(x, \text{Noun.sem}) \} \]
\[ PP \rightarrow P \cdot NP \quad \{ P.sem(NP.sem) \} \]
\[ \text{Nominal} \rightarrow \text{ADJ} \cdot \text{Nominal} \]
\[ \quad \{ \lambda z.\text{Nominal.sem}(x) \land \text{AM}(x, \text{ADJ.sem}) \} \]
\[ \text{Det} \rightarrow \text{“the”} \quad \{ \exists! \} \]
\[ \text{Det} \rightarrow \text{“a”} \quad \{ \exists \} \]
\[ \text{P} \rightarrow \text{“of”} \quad \{ \lambda x,y.\text{OF}(x,y) \} \]

— adapted from Jurafsky and Martin, 2000.
Example Lexicon

\[ V \rightarrow \text{“list”} \]
{\[ \lambda x, y. \exists e \text{ISA}(e, \text{listing}) \land \text{Actor}(e, y) \land \text{Object}(e, x) \]}

\[ V \rightarrow \text{“give”} \]
{\[ \lambda x, y. \exists e \text{ISA}(e, \text{giving}) \land \text{Actor}(e, y) \land \text{Object}(e, x) \]}

\[ N \rightarrow \text{“phone number”} \quad \{\text{phoneNumber}\} \]
\[ N \rightarrow \text{“restaurants”} \quad \{\text{restaurant}\} \]
\[ \text{Proper Noun} \rightarrow \text{“Rex”}\{\text{Rex}\} \]
\[ \text{ADJ} \rightarrow \text{“Italian”}\{\text{Italian}\} \]
Example *Mapping to intermediate form with complex terms*

\[
P(e,(\exists!x)(\phi(x))) \\
(\exists!x)(P(e,x) \land \phi(x))
\]

“List the Italian restaurants”:
\[
IMP((\exists e)(ISA(e,\text{listing}) \land \text{Actor}(e,\text{Computer}) \land \\
(\exists!z)(\text{Object}(e,z) \land \text{restaurant}(z) \land \\
\text{AM}(z,'\text{Italian'}))))
\]

“Give the phone number of Rex?”:
\[
IMP((\exists e)(ISA(e,\text{giving}) \land \text{Actor}(e,\text{Computer}) \land \\
(\exists!z)(\text{Object}(e,z) \land \text{phoneNumber}(z) \land \\
\text{OF}(z,'\text{Rex'}))))
\]
Example Translation Knowledge

IF

\[ IMP((\exists e)((ISA(e, \text{listing}) \lor ISA(e, \text{giving}) \lor \ldots) \land \text{Actor}(e, \text{Computer}) \land (\exists !x)(\psi(x)))) \]

THEN for each \( a \in Eval(\text{Translate}(\psi(x))) \)

\[ \text{DECL}\{x \mapsto a\}\psi(x) \]

Translation knowledge:

\[ \{x|\text{restaurant}(x)\} \rightarrow \{x.name|\text{Restaurant}(x)\}\]
\[ \{x|\text{phoneNumber}(x) \land \text{OF}(x, \$C_1)\} \rightarrow \{x.phone|\text{Restaurant}(x) \land x.name = \$C_1\}\]
\[ \{x|\text{restaurant}(x) \land \text{AM}(x, \$C_1)\} \rightarrow \{\text{Restaurant}(x) \land (\exists y)(\text{Type}(y) \land x.id = y.id \land y.type = \$C_1)\} \]
Transportable Systems

+ Only (part of) the lexicon and translation knowledge needs to be ported to new database
+ Evaluates open domain grammars
+ Bi-directionality — depending on the grammar
+ Ontology may mediate between query and DB

  – Statistical parsers may need to be retrained
  – Translation knowledge not always trivial to encode even with tools
  – Idioms and domain specific style must somehow be encoded
  – Sentence fragments and ill-formed input
Classifying (some of) the 20th century work

<table>
<thead>
<tr>
<th>‘database’</th>
<th>Syntax based systems</th>
<th>Semantic grammars</th>
<th>Transportable systems</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relational</td>
<td></td>
<td>RENDEZVOUS, EUFID, PLANES, Q/A</td>
<td>TEAM, TELI, JANUS</td>
</tr>
<tr>
<td>Prolog</td>
<td>CHAT</td>
<td></td>
<td>MASQUE, LOQUI</td>
</tr>
<tr>
<td>Semantic Network/Description Logic</td>
<td>LUNAR</td>
<td>LADDER</td>
<td>JANUS, ASK</td>
</tr>
</tbody>
</table>
Difficulties

- Linguistic/Conceptual coverage not obvious
  - users over estimate coverage
  - users under estimate coverage
  - users conflate linguistic and conceptual limitations
  - some empirical work suggests that linguistic limitations more serious

- Limited Context/Register handling
  - in informal register ‘and’ can mean ‘or’
  - users prefer brevity over grammatical correctness

- Inappropriate Medium

- Tedious/expensive configuration

- Symantex started with Q&A!, but now does virus protection

- Computational Linguistics went in a different direction:
  - Statistical/Probabilistic NLP
  - Lexical Semantics (POS tagging)
  - Information Extraction
  - ...
Why now?

- Lexical databases! (WordNet)
- Mature relational database technology
- Fast theorem provers
- Web-based evaluation
- Statistical parsers (modifier attachment)
- Ontologies
- Speech recognition always improving
- ...
Recent Work

- Microsoft English Query (1999)
  + authoring wizards
  - no paraphraser, no response generator, no update query support
  - failed...

- PRECISE (2003)
  reduces semantic analysis to a graph matching problem
  works for limited class of so called *semantically tractable queries*
  + requires very little configuration
  + leverages domain independent grammars
  + they put up a web interface!
  - not clear how to address more expressive queries
  - no paraphraser
  - they took away web interface!

- STEP (2004-?)
$D$: a dictionary of words (or word sequences) (e.g. $D = 'aardvark'$,...)

$W$: wh-words ‘who’, ‘what’,...

$F$: function words ‘a’, ‘the’, ‘in’...

$E = E_r \cup E_a \cup E_v$: a set of database elements (relations, attributes and values)

$N \subseteq E \times D$: a binary naming relation

$N_w \subseteq N$: a set of appropriate wh-words paired with relations and attributes,

$N_w \subseteq E_r \cup E_a \times W$.

$P$: part of relation $P(E, E') \subseteq E \times E$ (e.g. an attribute is part of a relation, a value is part of a attribute)
Precise(2003)

\( \text{Parse}(w_1, \ldots, w_m) \rightarrow (T, AT) \) where \( T \) is a set of tokens \( (T \subseteq D) \) and \( AT \) is a binary relation over \( T \).

Function words in \( w_1, \ldots, w_m \) are stripped from word sequence: \( F \cap T = \emptyset \)

\( AT \) denotes syntactic attachment and is obtained from a domain independent grammar (currently Charniak’s parser)

A matching \( M \) of \( T \) to \( E \) is a pairing of every token \( t \in T \) with some entity \( e \in E \) such that \( (e, t) \in N \).

\( M \) is valid if for all \( (t, e) \in M \):

if \( (t, e) \in M \) where \( e \in E_a \cup E_v \) then some \( (t', e') \in M \) where \( P(e, e') \) and \( AT(t, t') \)

\( W \) is semantically tractable iff there exists a valid matching for \( \text{Parse}(W) = (T, A) \) where there is at least one token \( t \in W \)

“Theorem 1. PRECISE is sound and complete for any semantically tractable question.”

Let’s evaluate this claim...
• Uses a phrasal lexicon:
  – a type of highly structured semantic grammar
  – tools help rapid authoring
  – customization of phrases
• Includes a query paraphraser
• Enables natural language based database updates
• Has captured 1000s of geography queries

Would like to say more, but don’t have the time...
Evaluation

- **Benchmark studies:**
  
  human built ‘gold standard’ of NL sentences/logical queries pairs (possibly obtained through "WOz" studies)
  
  + Can set a bar on the expressivity of queries
    
    (see Copeskate and Spark Jones, 1990)
    
    - Hard to trust accuracy reports

- **Task-based studies:**
  
  Compare performance of approaches for a given task
  
  + Randomized experiments (via web)
  
  - Lots of subjective interpretation of results
  
  - Requires full fledged prototype

- **Fielded systems studies:**
  
  + Web interface to satisfy real needs!
  
  + Capture real queries
  
  - Still looking for winner! How about BASEBALL?
  
  - Requires product level robustness
Conclusions

- NL interfaces to databases is an important, though currently understudied area
- It is critical to work over standard relational databases
- It is critical to paraphrase queries
- Dialectic between Semantic Grammars and Transportable Systems
- Advances of the past 15-20 years in computational linguistics/databases can impact this area
- STEP represents a comprehensive, semantic grammar based approach under active development