Incremental Query Generation

Laura Perez-Beltrachini* Claire Gardent‡ Enrico Franconi*

‡CNRS/LORIA

*Free University of Bozen-Bolzano

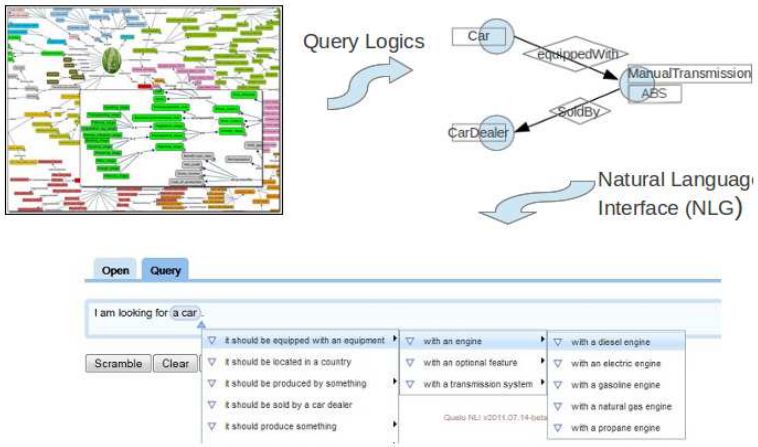
Ontology-based query tool: Quelo

[Franconi et al., 2010a, Franconi et al., 2011, Franconi et al., 2010b]

Efficient and user-friendly way of querying data sources.

- makes use of ontologies and reasoning services
- provides query manipulation operations and a NL interface

Users do not need to know about the KR language nor about the KB/DB structure

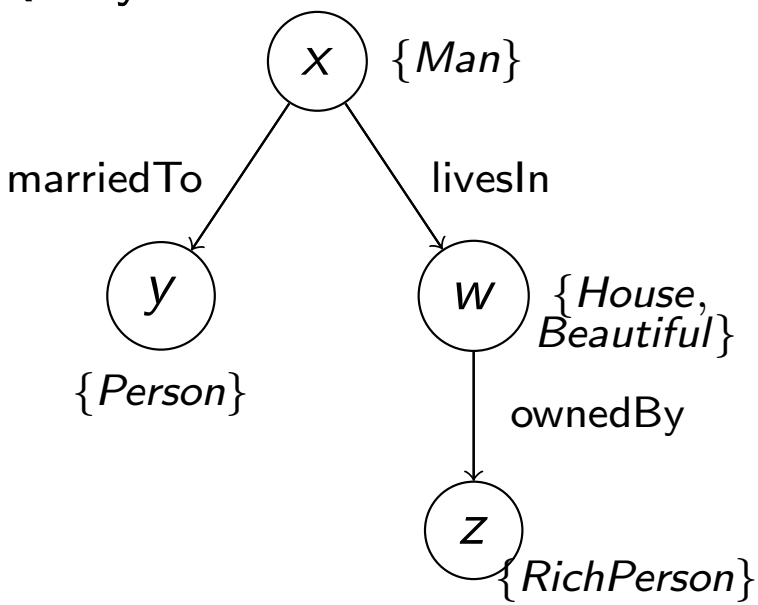


Query language, representation and operations

- Tree shaped conjunctive DL queries

$$
\begin{aligned}
S &::= C \mid \exists R.(S) \mid S \sqcap S \\
&\quad (\exists \textit{ limited existential restriction, } \sqcap \textit{ conjunction}) \\
R &::= r_i \textit{ for each } r_i \in Ro \\
C &::= c_i \textit{ for each } c_i \in Ac
\end{aligned}
$$

Query tree:



Operations:

- addCompatible(node, concept)
- addProperty(node, relation, concept)
- substitute(selection, concept)
- deletion(selection)

Quelo's Natural Language Interface (NLI)

Masks the composition of a formal query as the composition of English text (Conceptual Authoring).

At each point during the interactive query formulation process:

- computes all extensions of the current query that are consistent and non redundant (using automated reasoners)
- displays a NL description of these extensions

Quelo's Natural Langauge Generation (NLG) module

Goal: Improve fluency and generation flexiblity.

- Current template-based approach
  - provides a restricted set of syntactic constructions and uses ad-hoc methods
    - sequences of NP VP (slots filled in by concepts and relations) + ellision of repeated elements
  - not clear how to extend them for the verbalisation of query results
- Proposed grammar-based approach
  - allows for syntactic variability (e.g. relative clauses, PPs, etc.)
  - chart-based algorithm supports the incremental generation required by ontology-based querying

Outline of the talk

Incremental query generation

NLG architecture

Evaluation

Discussion

Text revisions

I am looking for **something** (initial query)

$\top$

I am looking for **a man** (substitute concept)

Man

I am looking for a young **man** (add compatible concept)

$Man \sqcap Young$

I am looking for a young **man who is married to a Person** (add relation)

$Man \sqcap Young \sqcap \exists isMarried.(Person)$

I am looking for a **young** married man (substitute selection)

$MarriedMan \sqcap Young$

I am looking for a married man (delete concept)

MarriedMan

Order constraints

$$Car \sqcap \exists runOn.(Diesel) \sqcap \exists equippedWith.(AirCond)$$

a. A car which runs on Diesel and is equipped with air conditioning

b. A car which is equipped with air conditioning and runs on Diesel

Constraints on the generation of DL queries

Input: current DL query Q (i.e. query tree) update U
Output: NL verbalisation of Q incorporating U

- support the modifications, deletions and additions required by incremental processing
- the query revisions should minimally effect the linear order of the NL query

The NLG Architecture

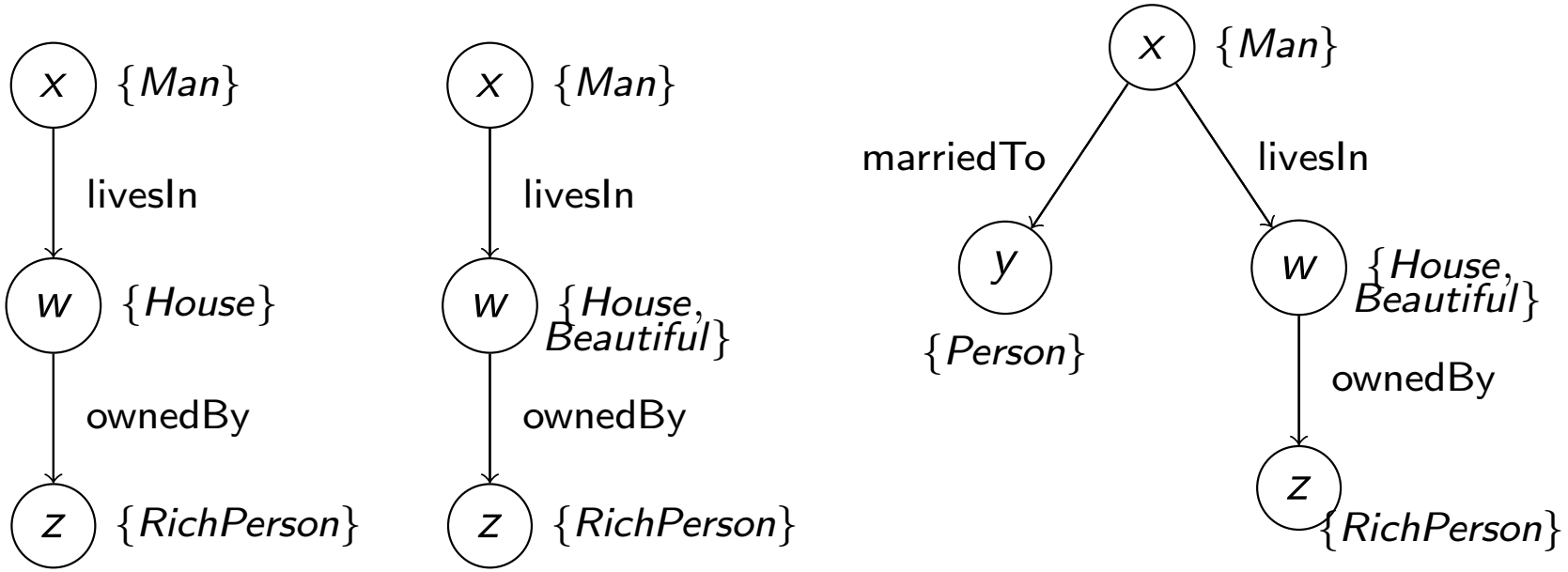
Document planning: linearises the input query and partitions the input into sentence size chunks

Surface realisation: maps each sentence size $\mathcal{L}$ formula into a sentence.

Referring expression generator: verbalises NPs.

Query linearisation

Document planning[Dongilli, 2008, Franconi et al., 2010a]



Man marriedTo Person livesIn House Beautiful ownedBy RichPeron

Man(m)[0] marriedTo(m,p)[1] Person(p)[2] livesIn(p,h)[3] House(h)[4] Beautiful(h)[5] ownedBy(h,r)[6] RichPerson(r)[7]

Content segmentation

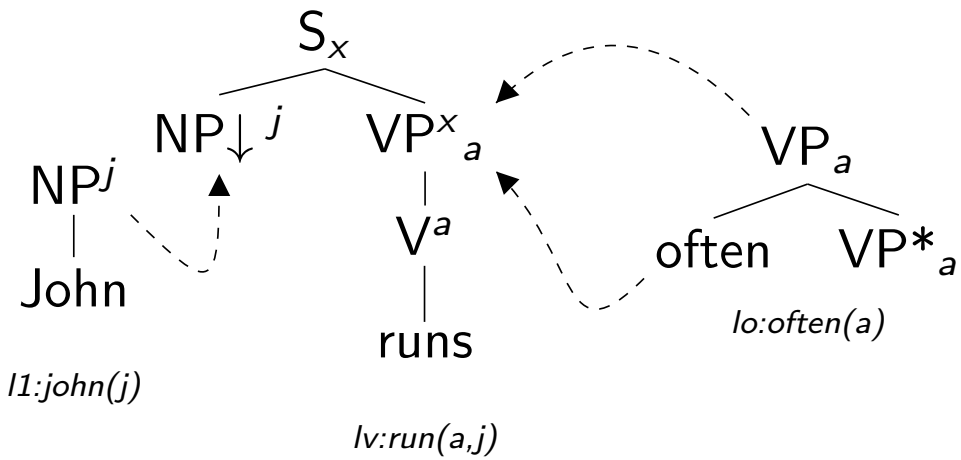
Document planning

Given a linearised query q, the document planner uses some heuristics based on the number and the types of relations/concepts present in q to output a sequence of sub-formulae each of which will be verbalised as a sentence.

Grammar based Surface Realisation

- Difficulty to find corpora containing the queries an their possible increments
- Need to produce query verbalisations for ontologies of any domain
- SemTAG naturally supports conceptual authoring
  - It systematically relates text, syntax and semantics
- Automatic lexicon extraction (map concept/relations into TAG trees [Trevisan, 2010])
- Tabular Algorithm
  - Efficient (avoids recomputation of intermediate structures)
  - Simple implementation of revisions (addition, deletion, substitution) operations
- Beam search
  - Cost function to enforce constituent ordering preferences

Feature-based Lexicalised Tree Adjoining Grammar (FB-LTAG) equipped with semantics



l1:named(j john), lv:run(a,j), lv:often(a)

Incremental chart-based realisation

C, the current chart. A, an empty agenda.

Add concept or property *X*: the trees selected by X are added to A and tried for combination with the elements of C.

Substitute selection *X* with *Y*: all chart items derived from a tree selected by X are removed from the chart. Conversely, all chart items derived from a tree selected by Y are added to the agenda and tried for combination with the elements of C.

Delete selection *X*: all chart items derived from a tree selected by X are removed from C. Intermediate structures that had previously combined with a tree selected by X are moved to the agenda and the agenda is processed until generation halts.

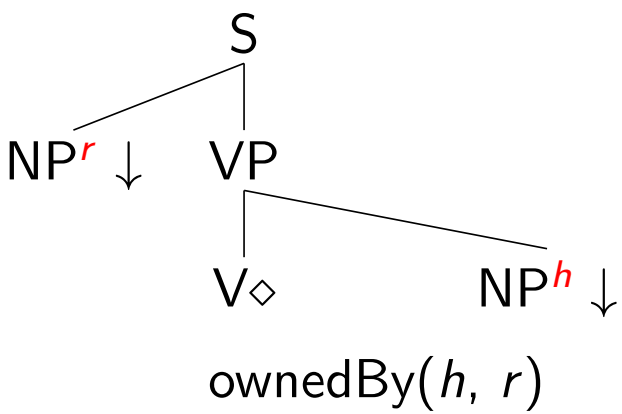
Beam search

Scoring Function *favors derivations with low word order cost and large semantic coverage.*

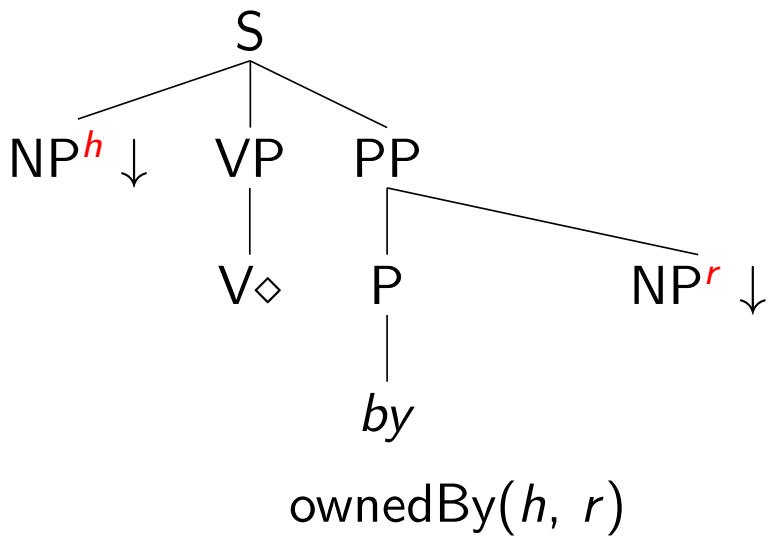
Word Order Cost = distance between actual position and required position (given by the linearised input)
Semantic Coverage = number of literals covered by derivation

Beam search (Ct'd)

House(h)[0] ownedBy(h,r)[1] RichPerson(r)[2]



× *The rich person owns a house.*



✓ *The house is owned by a rich person.*

Referring Expression Generation

Input: Sequence of phrase structure trees output by the surface realiser.

Uses heuristics to decide for each NP whether it should be verbalised as a pronoun, a definite or an indefinite NP.

Heuristics based on the linear order and the morpho-syntactic information contained in the phrase structure trees of the generated sentences.

Linearisation

- 4 series of queries $q_1 \cdots q_n$ where q_{i+1} is an increment of q_i
- 14 revisions in total
- encompass addition, deletion and substitution of possible operations at different points of the preceding query
- for all queries, the word order produced by the generator matches the linearisation of the DL query.

Assessing Quelo template-based queries

Fluency and clarity

41 queries capturing different combinations of concepts and relations

8 raters

50% of the queries are rated as disfluent
10% of the queries are rated as unclear

Free Comments: too repetitive, lacks aggregation

Comparing template vs grammar -based queries

Fluency and clarity

10 raters, 14 query pairs built from two ontologies (cars, universities)

	Fluency	Clarity
Grammar	**19.76**	6.87
Templates	7.2	8.57

Portability

- General, domain independent, grammar + Automatically extracted lexicon (cf. [Trevisan, 2010]).
- Lexicon extraction tested on 200 ontologies. Coverage: 85% of the ontology relations (12000 relns, 13 templates)
- 40 queries on 5 ontologies (cinema, wines, human abilities, assistive devices, ecommerce). Coverage 87%

Conclusions and future work

- Previous approach uses ad hoc generation algorithm based on templates
- Tabular algorithm naturally supports the definition of an incremental algorithm for query verbalisation
- The grammar based approach generates queries that are better accepted by human users

Conclusions and future work (Ct'd)

- Improve fluency, clarity (lexicon extraction, SR ranking)
- Use existing system to build a parallel corpus (DL/NL query) and train
  - a joint model of input segmentation, surface realisation and referring expression generation
  - a ranking module (to guide beam search)

Thank you!

References I

Dongilli, P. (2008).

Natural language rendering of a conjunctive query.
KRDB Research Centre Technical Report No. KRDB08-3). Bozen, IT: Free University of Bozen-Bolzano, 2:5.

Franconi, E., Guagliardo, P., and Trevisan, M. (2010a).

An intelligent query interface based on ontology navigation.
In *Workshop on Visual Interfaces to the Social and Semantic Web, VISSW*, volume 10. Citeseer.

Franconi, E., Guagliardo, P., and Trevisan, M. (2010b).

Quelo: a NL-based intelligent query interface.
In *Pre-Proceedings of the Second Workshop on Controlled Natural Languages*, volume 622.

Franconi, E., Guagliardo, P., Trevisan, M., and Tessaris, S. (2011).

Quelo: an Ontology-Driven Query Interface.
In *Description Logics*.

Trevisan, M. (2010).

A Portable Menuguided Natural Language Interface to Knowledge Bases for Querytool.
PhD thesis, Master's thesis, Free University of Bozen-Bolzano (Italy) and University of Groningen (Netherlands).