NLP-SQL CONVERTER

A STUDY ON THE VARIOUS METHODS

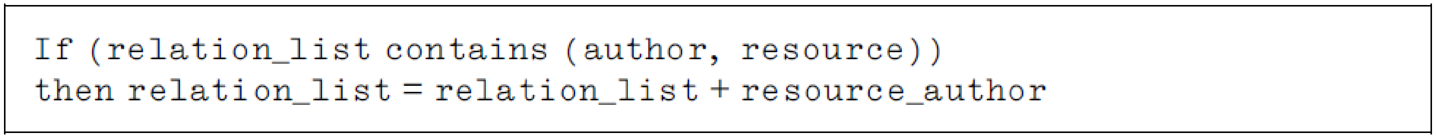
FUZZY ADVENTURE / SAP

VARIOUS ARCHITECTURES BEING USED FOR SQL CONVERTER

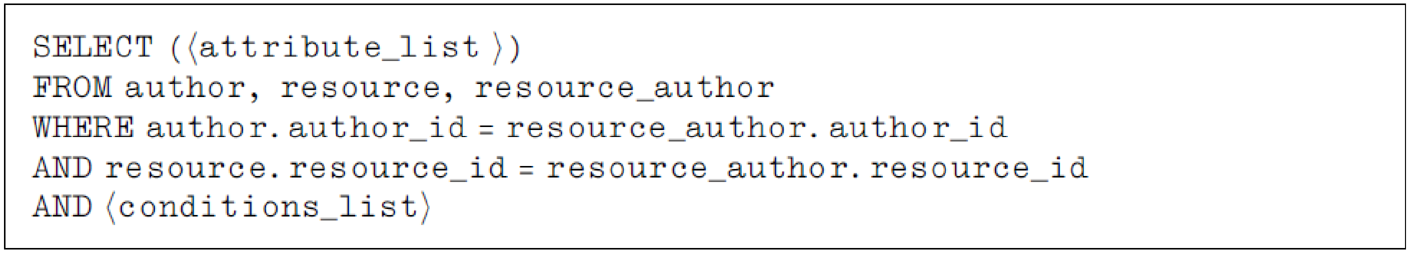
- Pattern Matching Systems
  - Our current SQL module
  - Works for small number of questions
  - It’s shallowness ends up in bad failure (matching a word in a wrong meaning)
    - E.g. Titles of employees in Los Angeles: the system (takes ‘in’ as a part of postal code from Indiana state and the system thinks the question is about the employees working in Indiana)
  - Advantage: it usually returns an answer because it finds the matches for all keywords (search 360, crash and burn in our system)
      - <att> of <obj.>
      - <att> of <obj.>of <obj.>
      - <action verb> obj. <att.Value>

VARIOUS ARCHITECTURES BEING USED FOR SQL CONVERTER

- Hybrid of Pattern Matching and Relation Rules:
  - Relations rules are meant to take into account dependencies between relations; and indicate the general pattern for mapping a user input to an SQL query.
  - The relation rules determine the list of relation and the list of conditions in the final SQL query
  - Our NLIDB(Semantic Net)+SQl modules [1]



```
If (relation_list contains (author, resource))
then relation_list = relation_list + resource_author
```



```
SELECT (⟨attribute_list ⟩)
FROM author, resource, resource_author
WHERE author.author_id = resource_author.author_id
AND resource.resource_id = resource_author.resource_id
AND ⟨conditions_list⟩
```

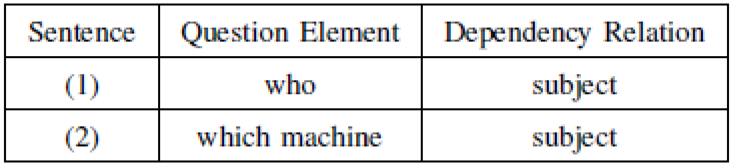
VARIOUS ARCHITECTURES BEING USED FOR SQL CONVERTER

- Syntax-Based Systems
  - The question is parsed syntactically and the parse tree is mapped directly to an expression in dB query.
  - It is difficult to devise the mapping rules that can do the mapping directly.
  - For more information please refer to [2].

VARIOUS ARCHITECTURES BEING USED FOR SQL CONVERTER

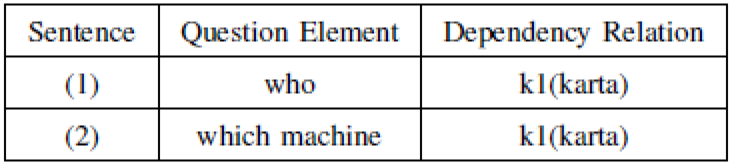
- Semantic Grammar Systems
  - This method also uses the parse trees for the mapping but it includes the semantic grammars as well.
  - Semantic knowledge should be hard-wired in the system and therefore it’s difficult to port these systems to other knowledge domain.
    - E.g. Computational Paninian Grammar Framework [3]
    - In our system who is matched to “employee”.

DEPENDENCY ANALYSIS



Sentence	Question Element	Dependency Relation
(1)	who	subject
(2)	which machine	subject

CPG BASED DEPENDENCY ANALYSIS

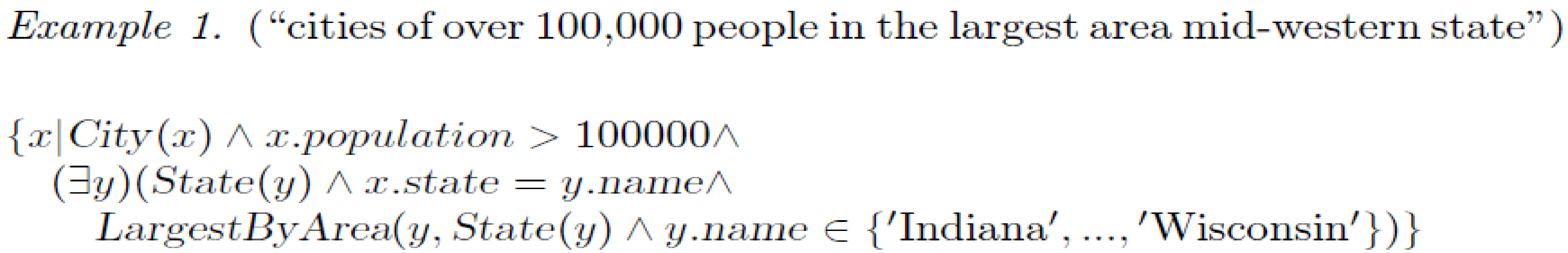


Sentence	Question Element	Dependency Relation
(1)	who	k1(karta)
(2)	which machine	k1(karta)

VARIOUS ARCHITECTURES BEING USED FOR SQL CONVERTER

- Intermediate Representation Languages
  - The question first gets converted to a logical query.
  - Logical query then generates the dB query

  - E.g. Using Combinatory Categorial Grammar (CCG) , Lambda Calculus
    - Paper: Giordani, Alessandra et al., “Semantic Mapping Between Natural Language Questions and SQL Queries via Syntactic Pairing”, [4,5]



Example 1. (“cities of over 100,000 people in the largest area mid-western state”)

$$\{x|City(x) \wedge x.population > 100000 \wedge \\ (\exists y)(State(y) \wedge x.state = y.name \wedge \\ LargestByArea(y, State(y) \wedge y.name \in \{'\mathrm{Indiana}', \ldots, '\mathrm{Wisconsin}'\})\}$$

VARIOUS ARCHITECTURES BEING USED FOR SQL CONVERTER

- E.g. Using LIL(Logic Intermediate Language):

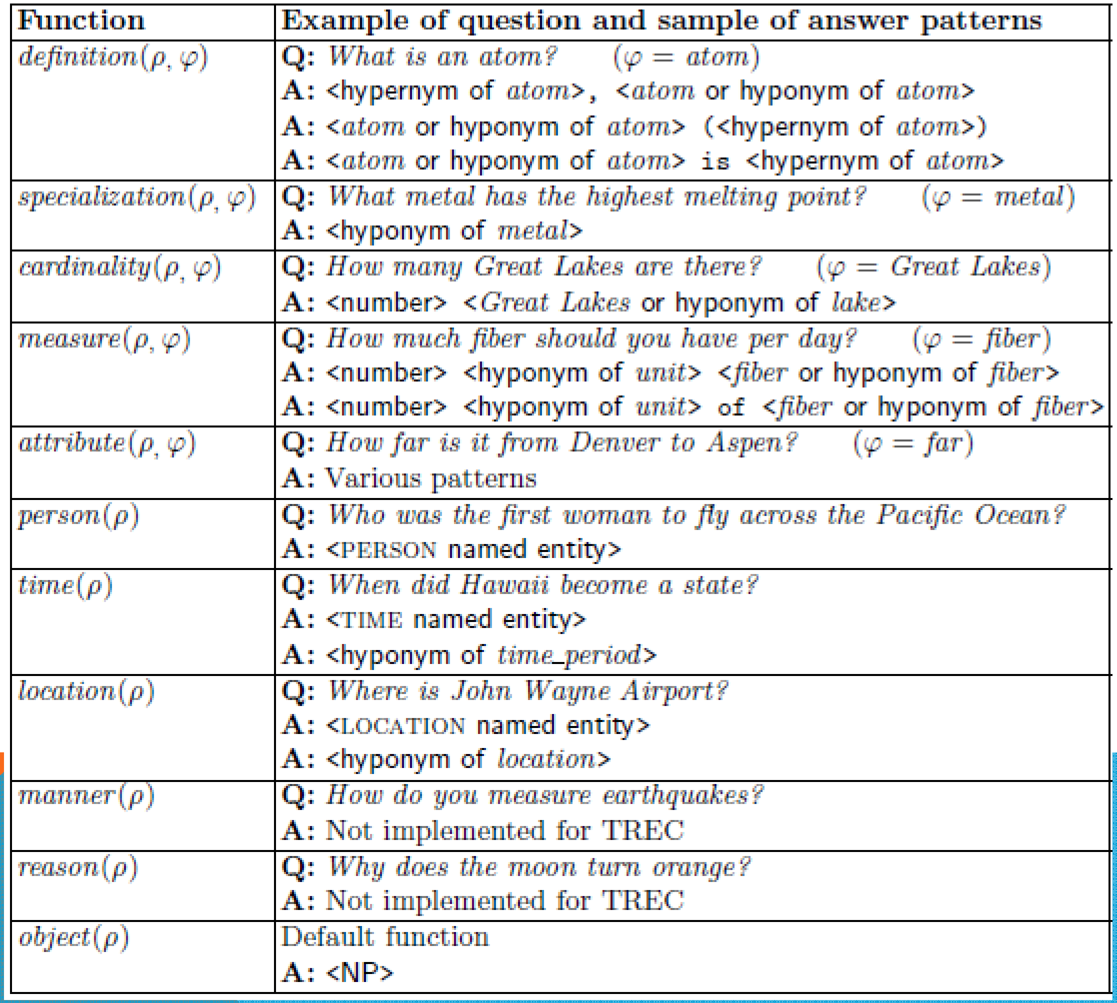
The question “which hotels do have sauna?” is translated to the following LIL expression: [6]

hotel(X), have(X,sauna), which is then transformed into SQL, for example:

```
SELECT HOTEL.NAME
FROM HOTEL
WHERE HOTEL.QT_SAUNA>0.
```

VARIOUS ARCHITECTURES BEING USED FOR SQL CONVERTER

A Function-Based Question Answering System, [7]



Function	Example of question and sample of answer patterns
definition(ρ, φ)	**Q:** *What is an atom?* $(\varphi = atom)$ **A:** <hypernym of *atom*>, <*atom* or hyponym of *atom*> **A:** <*atom* or hyponym of *atom*> (<hypernym of *atom*>) **A:** <*atom* or hyponym of *atom*> is <hypernym of *atom*>
specialization(ρ, φ)	**Q:** *What metal has the highest melting point?* $(\varphi = metal)$ **A:** <hyponym of *metal*>
cardinality(ρ, φ)	**Q:** *How many Great Lakes are there?* $(\varphi = Great\ Lakes)$ **A:** <number> <*Great Lakes* or hyponym of *lake*>
measure(ρ, φ)	**Q:** *How much fiber should you have per day?* $(\varphi = fiber)$ **A:** <number> <hyponym of *unit*> <*fiber* or hyponym of *fiber*> **A:** <number> <hyponym of *unit*> of <*fiber* or hyponym of *fiber*>
attribute(ρ, φ)	**Q:** *How far is it from Denver to Aspen?* $(\varphi = far)$ **A:** Various patterns
person(ρ)	**Q:** *Who was the first woman to fly across the Pacific Ocean?* **A:** <PERSON named entity>
time(ρ)	**Q:** *When did Hawaii become a state?* **A:** <TIME named entity> **A:** <hyponym of *time_period*>
location(ρ)	**Q:** *Where is John Wayne Airport?* **A:** <LOCATION named entity> **A:** <hyponym of *location*>
manner(ρ)	**Q:** *How do you measure earthquakes?* **A:** Not implemented for TREC
reason(ρ)	**Q:** *Why does the moon turn orange?* **A:** Not implemented for TREC
object(ρ)	Default function **A:** <NP>

REFERENCES:

1- Stratica, Niculae et al., ‘Using semantic templates for a natural language interface to the CINDI virtual library,’ Elsevier, 2004.

2- Giordani, Alessandra et al., ‘Semantic Mapping Between Natural Language Questions and SQL Queries via Syntactic Pairing,’ Via Sommarive 14, 38100 POVO (TN) – Italy.

3- Gupta, Abhijeet et al., ‘A Novel Approach Towards Building a Portable NLIDB System Using the Computational Paninian Grammar Framework’.

4- Minock Michael, ‘C-PHRASE: A system for building robust natural language interfaces to databases,’ Elsevier, 2009.

5- Traat, Maarika et al., ‘Unicational Combinatory Categorial Grammar: Combining Information Structure and Discourse Representations’.

6- P. Filipe, Porfírio et al., ‘Databases and Natural Language Interfaces,’ 1949-014 Lisboa, Portugal.

7- Plamondon, Luc, ‘QUANTUM: A Function-Based Question Answering System’.