Natural Language Data Management and Interfaces

Recent Development and Open Challenges

Yunyao Li
IBM Research - Almaden

Davood Rafiei
University of Alberta

SIGMOD PODS 2017 Chicago

“If we are to satisfy the needs of casual users of data bases, we must break through the barriers that presently prevent these users from freely employing their native languages"

Ted Codd, 1974

Employing Native Languages

- As data for describing things and relationships
  - Otherwise a huge volume of data will end up outside databases

- As an interface to databases
  - Otherwise we limit database use to professionals

Outline

- Natural Language Data Management
- Natural Language Interfaces for Databases
- Open Challenges and Opportunities

Natural Language Data Management

Outline of Part I

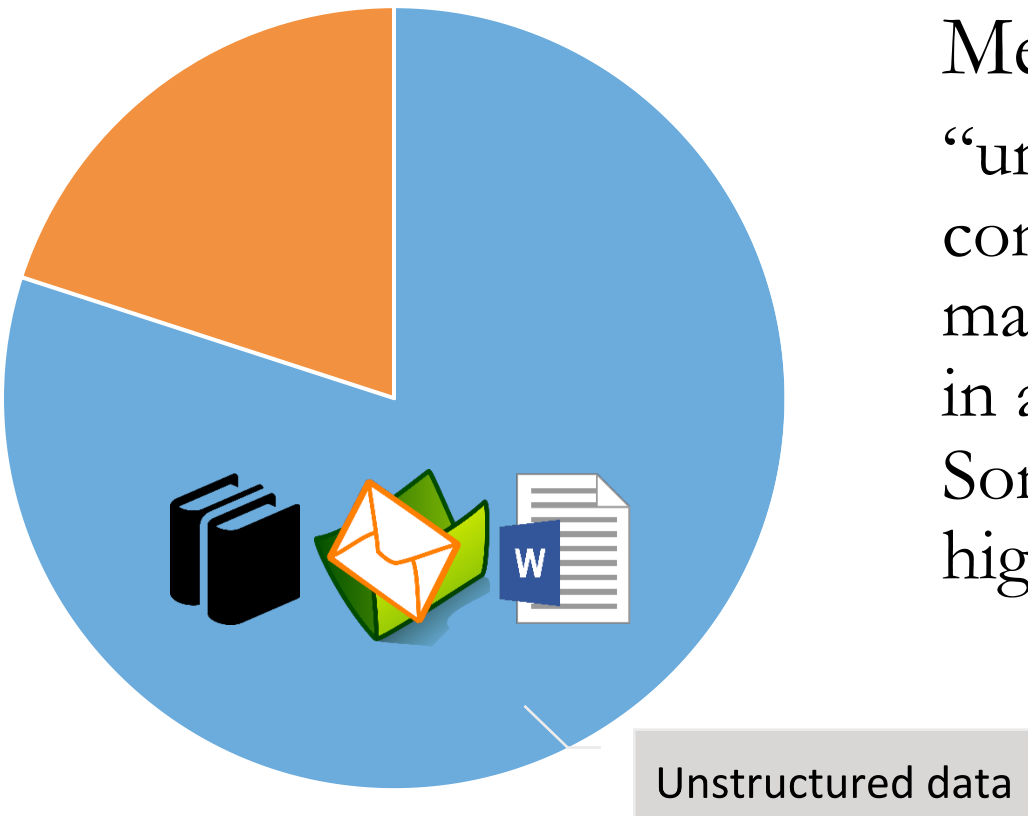
- The ubiquity of natural language data
  - A few areas of application
  - Challenges
- Areas of progress
  - Querying natural language text
  - Transforming natural language text
  - Integration

The Ubiquity of Natural Language Data

Data Domains

- Corporate data
- Scientific literature
- News articles
- Wikipedia

Corporate Data



Merril Lynch rule

"unstructured data comprises the vast majority of data found in an organization. Some estimates run as high as 80%."

Scientific Literature

Impact of less invasive treatments including sclerotherapy with a new agent and hemorrhoidopexy for prolapsing internal hemorrhoids.

Tokunaga Y, Sasaki H. (Int Surg. 2013)

Abstract

Abstract Conventional hemorrhoidectomy is applied for the treatment of prolapsing internal hemorrhoids. Recently, less-invasive treatments such as sclerotherapy using aluminum potassium sulphate/tannic acid (ALTA) and a procedure for prolapse and hemorrhoids (PPH) have been introduced. We compared the results of sclerotherapy with ALTA and an improved type of PPH03 with those of hemorrhoidectomy. Between January 2006 and March 2009, we performed hemorrhoidectomy in 464 patients, ALTA in 940 patients, and PPH in 148 patients with second- and third-degree internal hemorrhoids according to the Goligher's classification. The volume of ALTA injected into a hemorrhoid was 7.3 ± 2.2 (mean ± SD) mL. The duration of the operation was significantly shorter in ALTA (13 ± 2 minutes) than in hemorrhoidectomy (43 ± 5 minutes) or PPH (32 ± 12 minutes). Postoperative pain, requiring intravenous pain medications, occurred in 65 cases (14%) in hemorrhoidectomy, in 16 cases (1.7%) in ALTA, and in 1 case (0.7%) in PPH. The disappearance rates of prolapse were 100% in hemorrhoidectomy, 96% in ALTA, and 98.6% in PPH. ALTA can be performed on an outpatient basis without any severe pain or complication, and PPH is a useful alternative treatment with less pain. Less-invasive treatments are beneficial when performed with care to avoid complications.

Treatment

No of patients tries on

Duration

News Articles

April 25, 2017 12:48 pm

Loonie hits 14-month low as softwood lumber duties expected to impact jobs

By Ross Marowits The Canadian Press

MONTREAL – The loonie hit a 14-month low on Tuesday at 73.60 cents, the lowest level since February 2016.

The U.S. Commerce Department levied countervailing duties ranging between 3.02 and 24.12 per cent on five large Canadian producers and 19.88 per cent for all other firms effective May 1. The duties will be retroactive 90 days for J.D. Irving and producers other than Canfor, West Fraser, Resolute Forest Products and Tolko.

Anti-dumping duties to be announced June 23 could raise the total to as much as 30 to 35 per cent.
25,000 jobs will eventually be hit, including 10,000 direct jobs and 15,000 indirect ones tied to the sector
Dias anticipates that.

Event
Triggering event
Following events expected

Wikipedia

- 42 million pages
- Only 2.4 million infobox triplets
- Lots of data not in infobox

44th President of the United States	
In office January 20, 2009 – January 20, 2017	
Vice President	Joe Biden
Preceded by	George W. Bush
Succeeded by	Donald Trump
United States Senator from Illinois	
In office January 3, 2005 – November 16, 2008	
Preceded by	Peter Fitzgerald

Obama was hired in Chicago as director of the Developing Communities Project, a church-based community organization originally comprising eight Catholic parishes in Roseland, West Pullman, and Riverdale on Chicago's South Side.

...

In 1991, Obama accepted a two-year position as Visiting Law and Government Fellow at the University of Chicago Law School to work on his first book.

...

From April to October 1992, Obama directed Illinois's Project Vote, a voter registration campaign...

Community QA

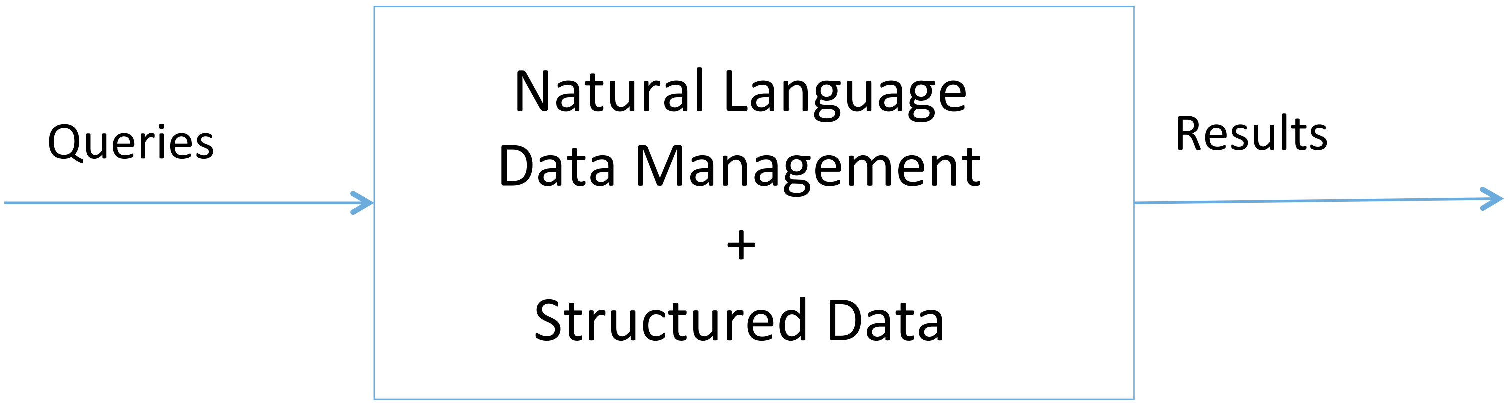
- Services such as Yahoo answers, Stack Overflow, AnswerBag, …
- Data: question and answer pairs
- Want answers to new queries

Q: How to fix auto terminate mac terminal

Two StackOverflow pages returned by Google

- osx - How do I get a Mac ".command" file to automatically quit after running a shell script?
- OSX - How to auto Close Terminal window after the "exit" command executed.

Vision


Queries
Natural Language
Data Management
+
Structured Data
Results

Challenges

Challenge – Lack of Schema

- The scientific article shown earlier contains structured data (as shown) but hard to query due to the lack of schema

treatment	patientCnt	duration	noOfPatients	disappearanceRate
sclerotherapy with ALTA	940	13+-2	16	96
PPH03	148	32+-12	1	98.6
hemorrhoidectomy	484	43+-5	65	100

Challenge - Opacity of References

- Anaphora
  - "Joe did not interrupt Sue because he was polite"
  - "the lion bit the gazelle, because it had sharp teeth"
- Ambiguity of ids
  - Does "john" in article A refer to the same "john" in article B?
- Variations due to spatiotemporal differences
  - "police chief" is ambiguous without a spatiotemporal anchor

Challenge - Richness of Semantics

- Semantic relations
  - crow $\subseteq$ bird; bird $\cap$ nonbird= {};
    bird $\cup$ nonbird=U
- Pragmatics
  - The meanning depends on the context
  - E.g. “Sherlock saw the man with binoculars”
- Textual entailment
  - “every dog danced” $\mapsto$ “every poodle moved”

Challenge - Correctness of Data

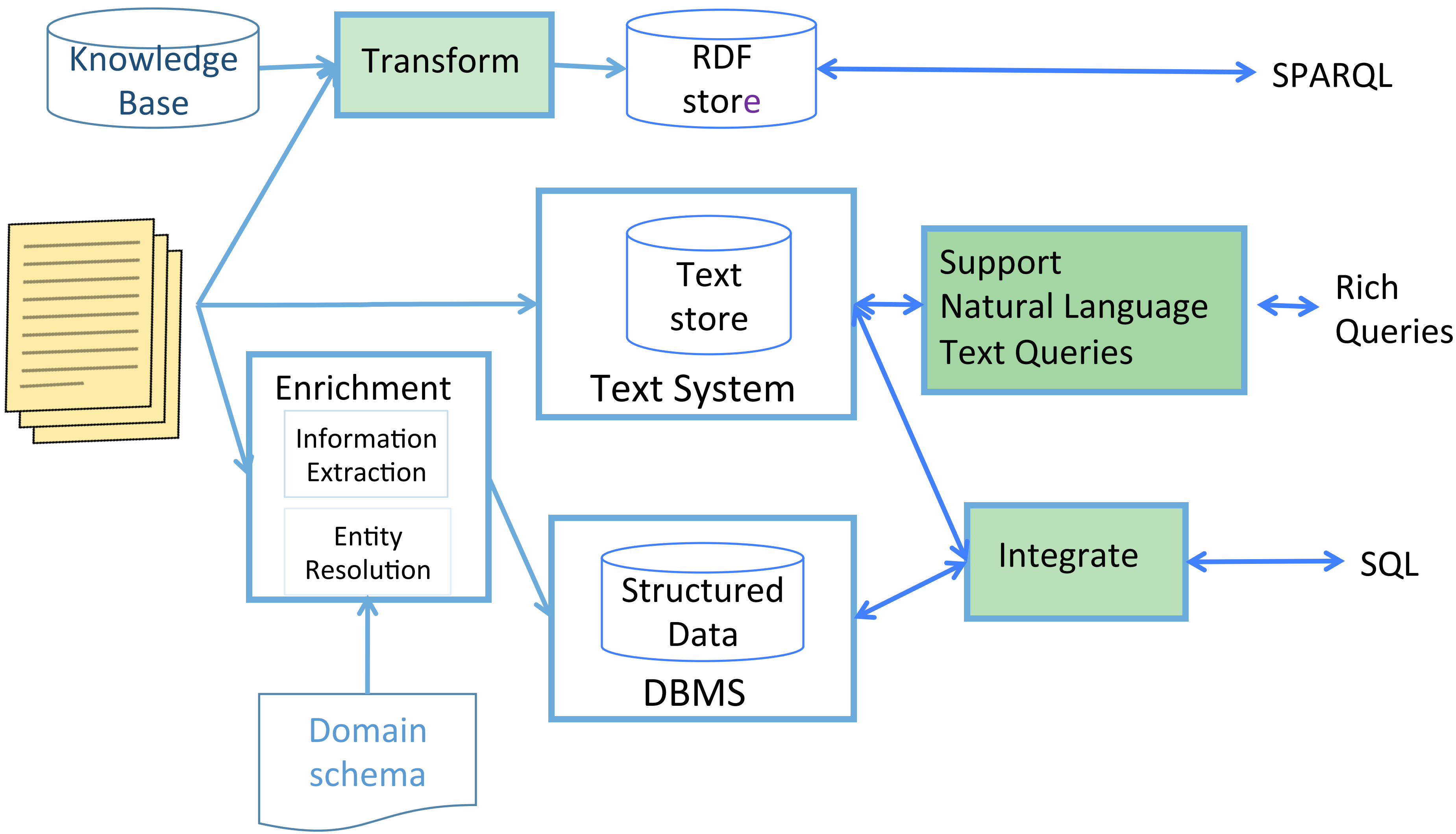
- Incorrect or sarcastic
  - “Vladimir Putin is the president of the US”
- Correct at some point in time (but not now)
  - “Barack Obama is the president of the US”
- Correct now
  - “Donald Trump is the president of the US”
- Always correct
  - “Barack Obama is born in Hawaii”
  - “Earth rotates around the sun”

Natural Language Data

- Text
- Speech

Focus: natural language text

System Architecture



Progress

- Entity resolution
- Information extraction
- Question answering
- Reasoning

Not Covered

Progress

- Support natural language text queries (rich queries)
- Transform
- Integrate

Covered

Support Natural Language Text Queries

Approaches

- Boolean queries
- Grammar-based schema and searches
- Text pattern queries
- Tree pattern queries

Boolean Queries

Highlight that due to the variants in NL, BQ can be extremely complex

- TREC legal track 2006-2012
  - Retrieve documents as evidence in civil litigation

((memory w/2 loss) OR amnesia OR Alzheimer! OR dementia) AND (lawsuit! OR litig! OR case OR (tort w/2 claim!) OR complaint OR allegation!)

from TREC 09 Legal track

- Default search in Quicklaw and Westlaw
  - E.g.

memory /2 loss
memory /s loss

Boolean Queries (Cont.)

- Not much use of the grammar
  - Except ordering and term distance
- Research issues
  - Optimization
    - Selectivity estimation for boolean queries [Chen et al., PODS 2000]
    - String selectivity estimation [Jagadish et al., PODS 1999], [Chaudhuri et al., ICDE 2004]
  - Query evaluation [Broder et al., CIKM 2003]

PAT Expressions

[Saliminen & Tompa, Acta Lingusitica Hungarica 94]

- A set-at-a-time algebra for text
- Text normalization
  - Delimiters mapped to blank, lowercasing, etc.
- Searches make less use of grammar
  - Lexical: e.g. “joe”, “bo”..“jo”
  - Position: e.g. [20], shift.2 “2010”..“2017”
    - The last two characters of the matches
  - Frequency: e.g. signif.2 “computer”
    - Significant two terms that start with “computer” such as “computer systems”

Mind your Grammar [Gonnet and Tompa, VLDB 1987]

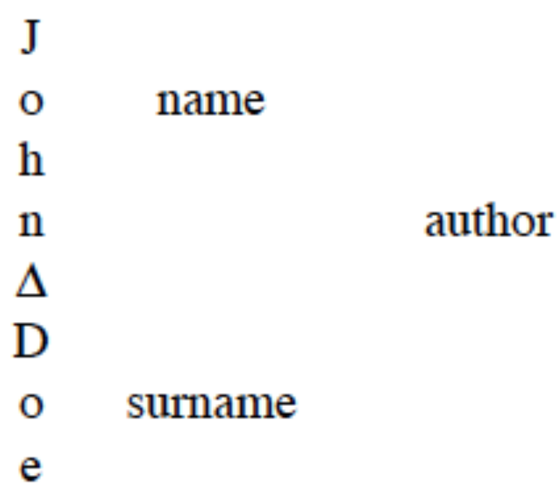
- Schema expressed as a grammar
- Studied in the context of *Oxford English Dictionary*

'man-trap, *n.*
A trap for catching men, *esp.* one for
1788 Wolcot (P. Pindar) *Peter's Pension* W
cock and hens. **1791** Boswell *Johnson* 20 M
we entered his garden of flowery eloquence,
Browning *Clive* 24 Did no writing on the wall '
transf. and *fig.* **1773** Goldsm. *Stoops to Con*

Word	Pos_tag	Pr_brit	Pr_us	Plurals	…
Man-trap	n				

Grammar-based Data

- The grammar (when known) allows data to be represented and retrieved
- Compared to relational data
  - Grammar ~ table schema
  - Parsed strings (p-strings) ~ table instance



Grammar-based Data (another context)

- Data wrapped in text and html formatting
  - Many ecommerce sites with back-end rel. data
- Grammar often simple
- Schema finding ~ grammar induction
  - Input: (a) html pages with wrapped data, (b) sample/tagged tuples
  - Output: a grammar (or a wrapper)

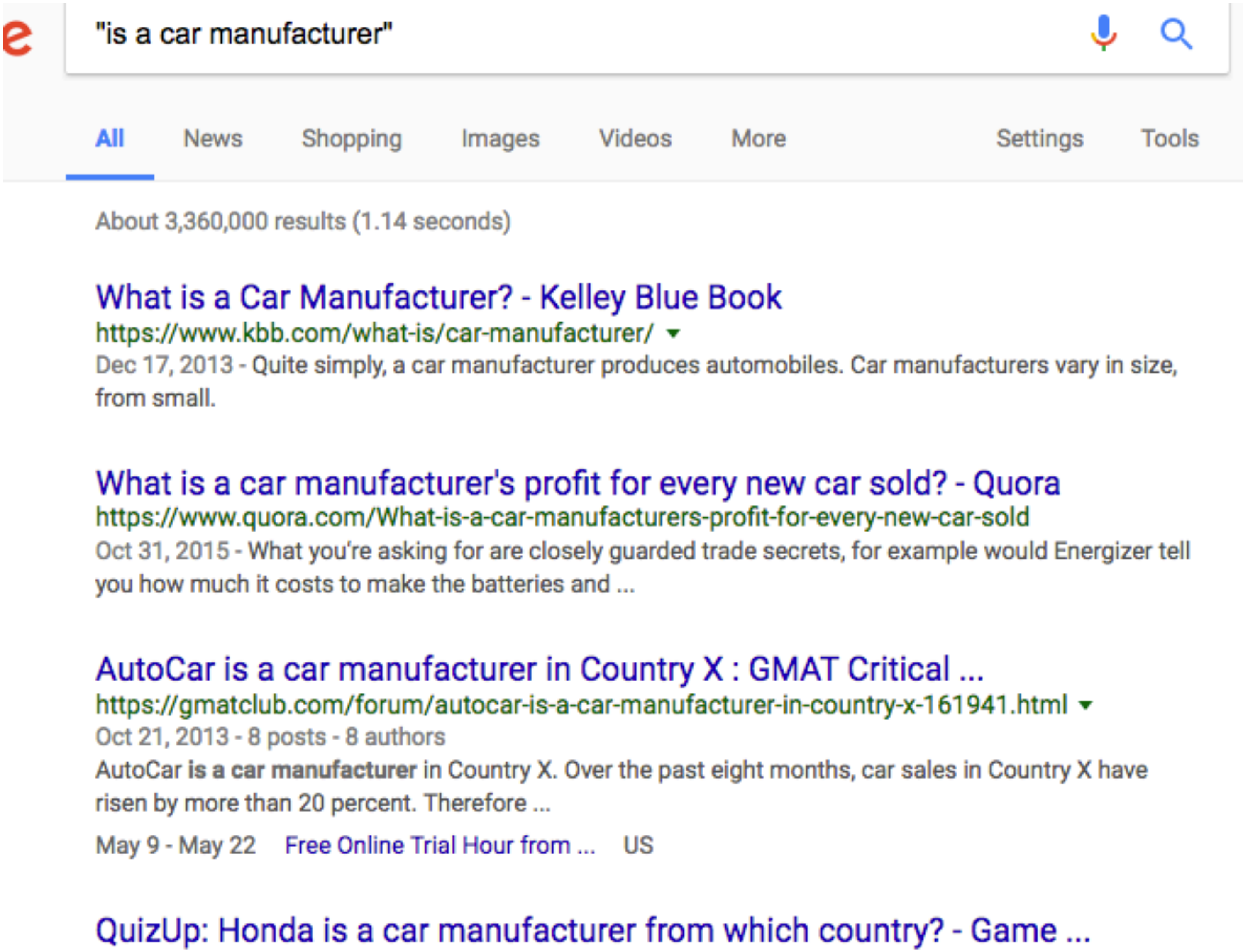
Grammar Induction

- Challenge: Regular grammars cannot be learned from positive samples only [Gold, Inf. Cont. 1967]
  - Many web pages use grammars that are identifiable in the limit (e.g. [Crescenzi & Mecca, J. ACM 2004])
- With natural language text
  - Context free production rules exist for good subsets
  - Not deterministic (multiple derivations per input)
  - The rules are usually complex, less uniform, and maybe ambiguous

Text Pattern Queries

- Text modeled as “a sequence of tokens”
- Data wrapped in text patterns
  - <name> was born in <year>
  - Also referred to as surface text patterns [Ravichandran and Hovy, ACL 2002]
- Queries ~ text patterns

Google Search: “is a car manufacturer”



DeWild [Li & Rafiei, SIGIR 2006, CIKM 2009]

- Query match short text (instead of a page)
- Result ranking
  - To improve "precision at k"
- Query rewritings

DeWild Query: % is a car manufacturer

Instance	Weight
general motors	0.216994
toyota	0.196666
hyundai	0.194849
ford	0.19083
gm	0.19083
audi	0.188238
honda	0.186772
daimler chrysler	0.160607

Rewriting Rules

- Hyponym patterns [Hearst, 1992]
  - X such as Y
  - X including Y
  - Y and other X
- Morphological patterns
  - X invents Y
  - Y is invented by X
- Specific patterns
  - X discovers Y
  - X finds Y
  - X stumbles upon Y

Rewriting Rules in DeWild

```
# nopos
(.+),? such as (.+)
such (.+) as (.+)
(.+),? especially (.+)
(.+),? including (.+)
->
$1 such as $2          && noun(,$1)
such $1 as $2          && noun(,$1)
$1, especially $2      && noun(,$1)
$1, including $2       && noun(,$1)
$2, and other $1       && noun(,$1)
$2, or other $1        && noun(,$1)
$2, a $1               && noun($1,)
$2 is a $1             && noun($1,)
```

```
#pos
N<([^<>]+)>N,? V<(\w+)>V by N<([^<>]+)>N
N<([^<>]+)>N V<is (\w+)>V by N<([^<>]+)>N
N<([^<>]+)>N V<are (\w+)>V by N<([^<>]+)>N
N<([^<>]+)>N V<was (\w+)>V by N<([^<>]+)>N
N<([^<>]+)>N V<were (\w+)>V by N<([^<>]+)>N
->
$3 $2 $1               && verb($2,,,)
$3 $2 $1               && verb(,$2,,)
$3 $2 $1               && verb(,,$2,)
$3 will $2 $1          && verb($2,,,)
$3 is going to $2 $1   && verb($2,,,)
$1 is $2 by $3         && verb(,,,$2)
$1 was $2 by $3        && verb(,,,$2)
$1 are $2 by $3        && verb(,,,$2)
```

noun(country, countries)

verb(go, goes, went, gone)

Queries in DeWild

- Text patterns with some wild cards
- E.g
    - % is the prime minister of Canada
    - % invented the light bulb
    - % invented %
    - % is a summer *blockbuster*

Indexing for Text Pattern Queries

- Method 1: Inverted index

Query: Canada population is %

34,480,00 -> …, <2,1,[10]>, …

is -> <1,5,[4,16,35,58,89]>, …. <2,1,[9]>, …

population -> … <2,1,[8]> <3,1,[10]>, …

Canada -> … <2,1,[7]>, …

docId tf offset list

Indexing for Text Pattern Queries (Cont.)

- Method 2: Neighbor index

[Cafarella & Etzioni, WWW 2005]

34,480,00 -> …, <2,1,[(10,is,-)]>, …

is -> …. <2,1,[(9,population,34,480,000)]>, …

population -> … <2,1,[(8,Canada,is)]>, …

Canada -> … <2,1,[(7,though,population)]>, …

Problems: (1) long posting lists e.g. for “is”, “and”, …

(2) join costs |#(query terms) - 1| * |post_list($term_i$)|

Indexing for Text Pattern Queries (Cont.)

- Method 3: Word Permuterm Index (WPI)

  [Chubak & Rafiei, CIKM 2010]

  - Based on Permuterm index [Garfield, JAIS 1976]
  - Burrows-wheeler transformation of text [Burrows & Wheeler, 1994]
  - Structures to maintain the alphabet and to access ranks

Word-level Burrows-wheeler transformation

- E.g. three sentences (lexicographically sorted)

  T = \$ Rome is a city \$ Rome is the capital of Italy \$ countries such as Italy \$ ~

- BW-transform
  - Find all word-level rotations of **T**
  - Sort rotations
  - The vector of the last elements is BW-transform

BW-transformation

1	$ Rome is a city $ Rome is the capital of Italy $ countries such as Italy $ **~**
2	$ Rome is the capital of Italy $ countries such as Italy $ ~ $ Rome is a **city**
3	$ countries such as Italy $ ~ $ Rome is a city $ Rome is the capital of **Italy**
4	$ ~ $ Rome is a city $ Rome is the capital of Italy $ countries such as **Italy**
5	Italy $ countries such as Italy $ ~ $ Rome is a city $ Rome is the capital **of**
6	Italy $ ~ $ Rome is a city $ Rome is the capital of Italy $ countries such **as**
7	Rome is a city $ Rome is the capital of Italy $ countries such as Italy $ ~ **$**
8	Rome is the capital of Italy $ countries such as Italy $ ~ $ Rome is a city **$**
9	a city $ Rome is the capital of Italy $ countries such as Italy $ ~ $ Rome **is**
10	as Italy $ ~ $ Rome is a city $ Rome is the capital of Italy $ countries **such**
11	capital of Italy $ countries such as Italy $ ~ $ Rome is a city $ Rome is **the**
12	city $ Rome is the capital of Italy $ countries such as Italy $ ~ $ Rome is **a**
13	countries such as Italy $ ~ $ Rome is a city $ Rome is the capital of Italy **$**
14	is a city $ Rome is the capital of Italy $ countries such as Italy $ ~ $ **Rome**
15	is the capital of Italy $ countries such as Italy $ ~ $ Rome is a city $ **Rome**
16	of Italy $ countries such as Italy $ ~ $ Rome is a city $ Rome is the **capital**
17	such as Italy $ ~ $ Rome is a city $ Rome is the capital of Italy $ **countries**
18	the capital of Italy $ countries such as Italy $ ~ $ Rome is a city $ Rome **is**
19	~ $ Rome is a city $ Rome is the capital of Italy $ countries such as Italy **$**

Traversing L backwards

i	L
1	~
2	city
3	Italy
4	Italy
5	of
6	as
7	$
8	$
9	is
10	such
11	the
12	a
13	$
14	Rome
15	Rome
16	capital
17	countries
18	is
19	$

$\mathrm{Prev}(i) = \mathrm{Count}[L[i]] + \mathrm{Rank}_{L[i]}(L,i)$

Number elements smaller than L[i], in L

Occurrences of L[i] in the range (L[1..i])

$\mathrm{Prev}(8) = \mathrm{Count}(\$) + \mathrm{Rank}_{\$}(L,8)$
$= 0 + 2 = 2$
The second $ is preceded by city in **T**

$\mathrm{Prev}(10) = \mathrm{Count}(\text{such}) + \mathrm{Rank}_{\text{such}}(L,10)$
$= 16 + 1 = 17$
such is preceded by countries in **T**

T = $ Rome is a city $ Rome is the capital of Italy $ countries such as Italy $ ~

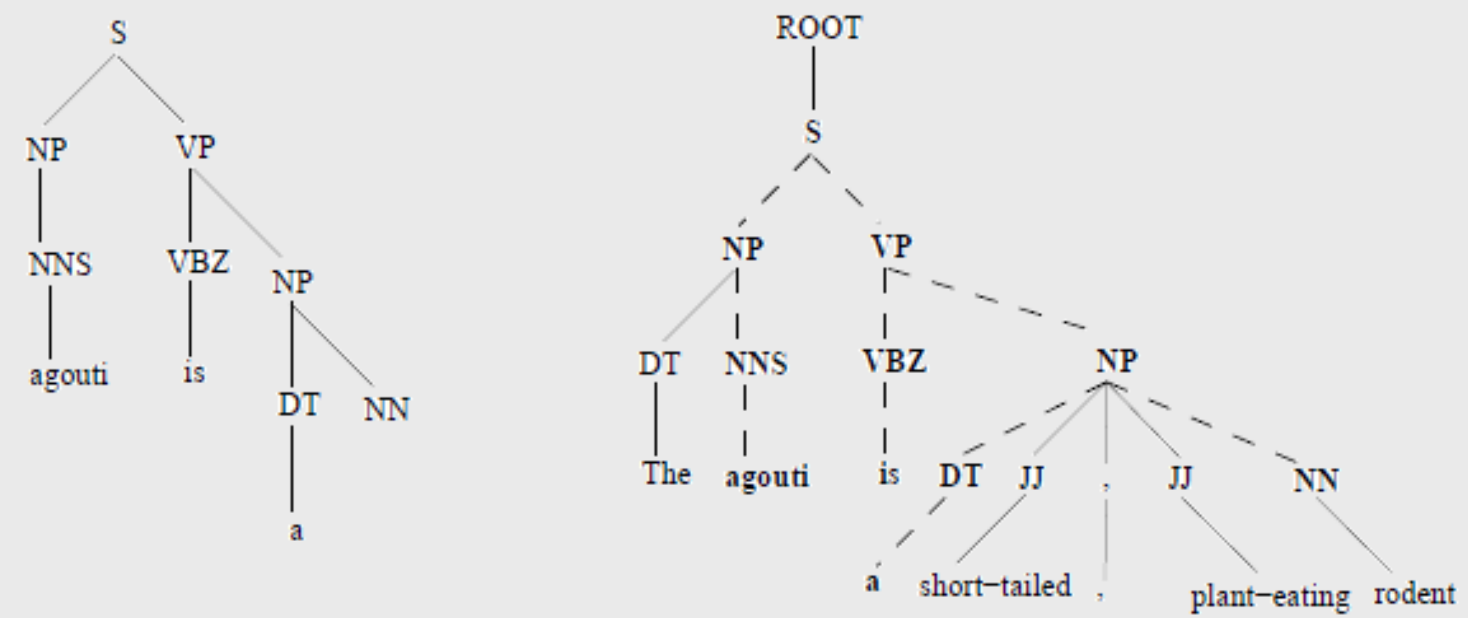
Prev(8) Prev(10)

Tree Pattern Queries

- Text often modeled as a set of “ordered node labeled tree”
  - Order usually correspond to the order of the words in a sentence
- Queries
  - Navigational axes: XPath style queries
    - E.g. find sentences that include `dog’ as a subject
  - Boolean queries
    - E.g. Find sentences that contain any of the words w1, w2 or w3.
  - Quantifiers and implications
  - Subtree searches

Subtree Searches

`What kind of animal is agouti?` **(TREC-2004 QA track)**



(a) parse tree of a sample query

(b) parse tree of a matching sentence

Approaches

- Literature on general tree matching
  - E.g. ATreeGrep [Shasha et al., PODS 2002]
  - Often do not exploit properties of Syntactically-Annotated Tree (SAT)
    - E.g. distinct labels on nodes
- Querying SATs
  - Work from the NLP community
    - E.g. TGrep2, CorpusSearch, Lpath
    - Scan-based, inefficient
  - Indexing unique subtrees

Indexing Unique Subtrees

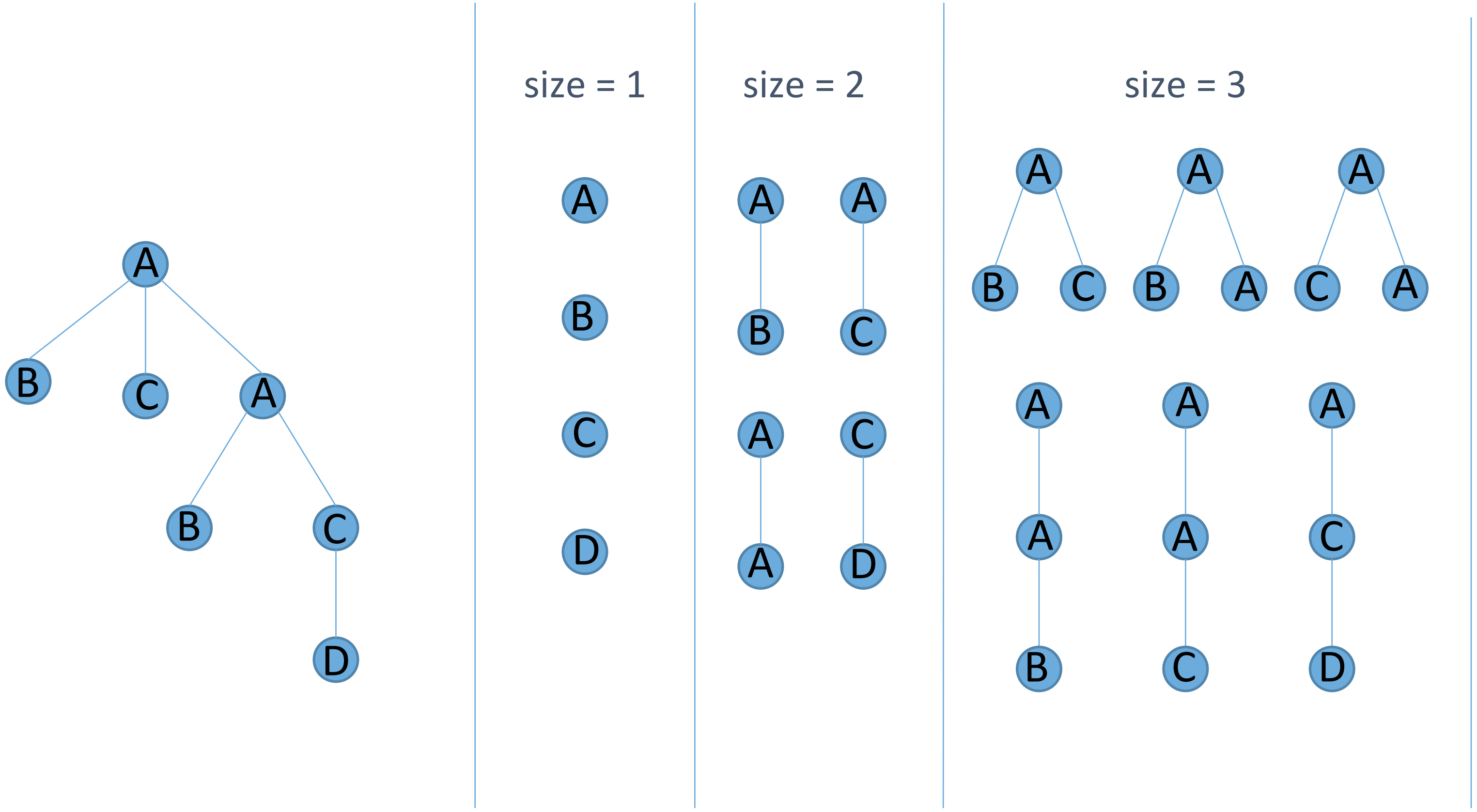
[Chubak & Rafiei, PVLDB 2012]

- Keys: unique subtrees of up to a certain size
- Posting lists: structural info. of keys

- Evaluation strategy: break queries into subtrees, fetch lists and join

- Syntactically annotated trees
  - Abundant frequent patterns → small number of keys
  - Small average branching factor → small number of postings

Example Subtrees



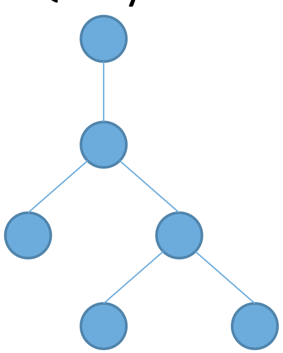
Subtree Coding

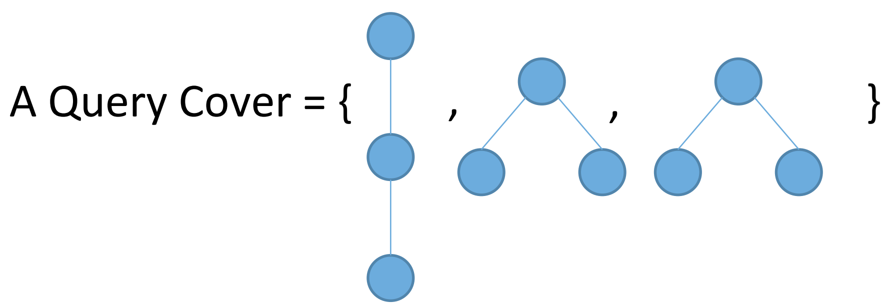
- Filter-based
  - Store only tid for each unique subtree in the posting list
  - No other structural information
- Subtree interval coding
  - Store pre, post and order values in a pre-order traversal (for containment rel.) and level (for parent-child rel.)
- Root split coding
  - Optimize the storage for subtree interval coding

Query Decomposition

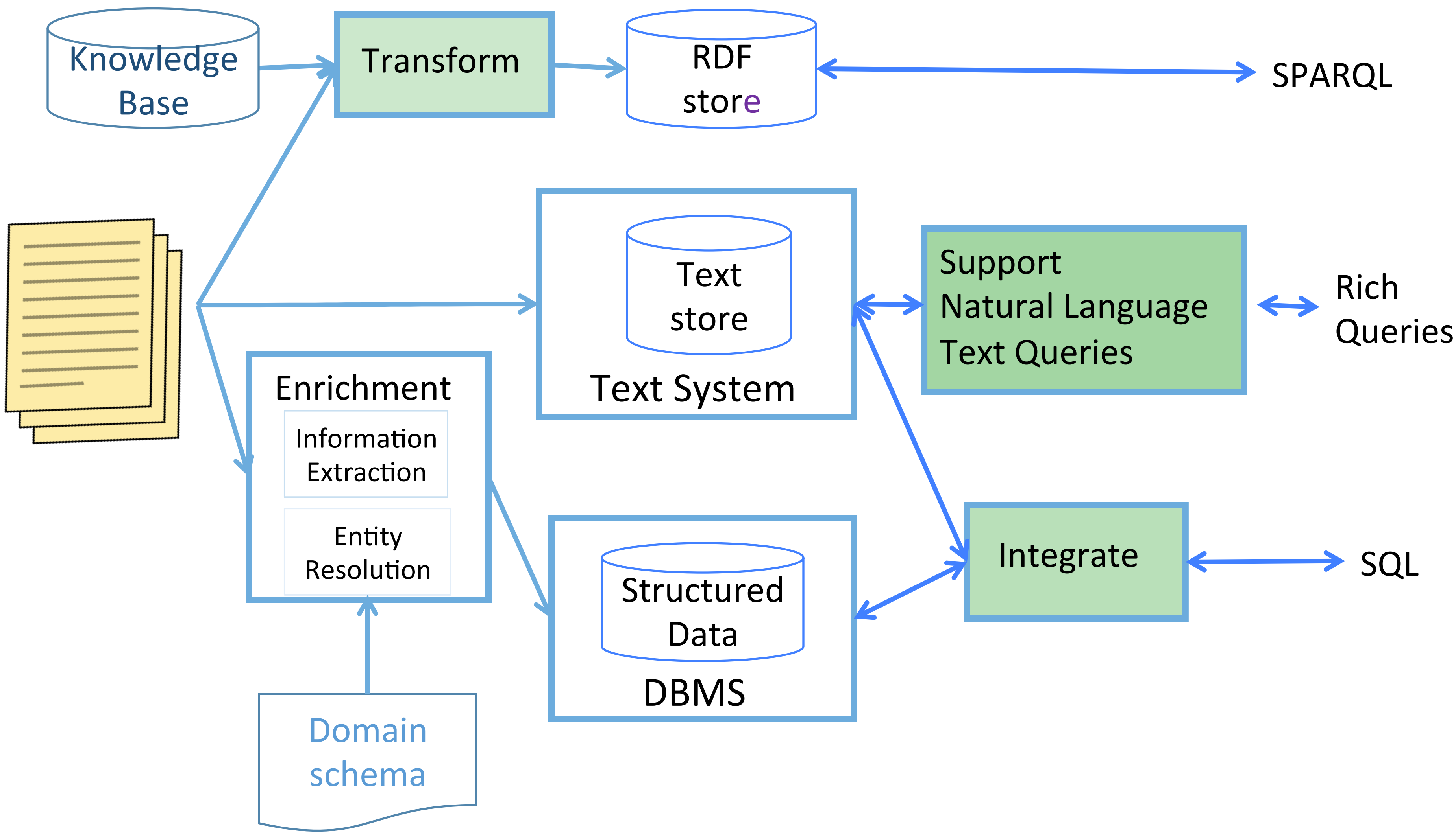
- Want an optimal cover to reduce the join cost
- Guarantee an optimal cover for filter-based and subtree interval coding
  - For subtrees of size 6 or less
- Bound the number of joins in a root split cover

Query





System Architecture



Transforming & Integrating Natural Language Data

Transforming Natural Language Data

- Transformation to a meaning representation (aka semantic parsing) such as
  - RDF triples
  - Other form of logical predicates

Transformed text is sufficient for querying (minimal loss)

Integrating Natural Language Data

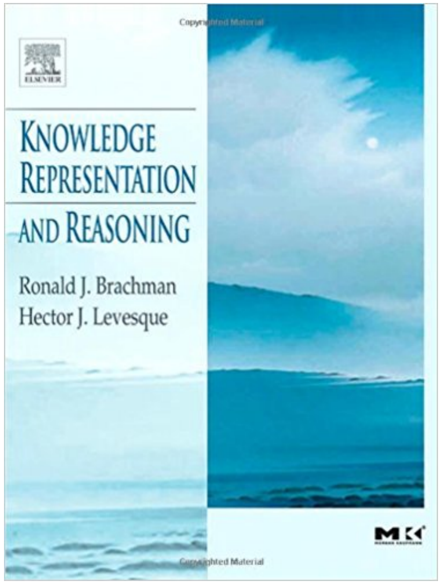
- Tight integration
  - Text is maintained by a relational system
- Lose integration
  - Text is maintained by a text system

Transforming Natural Language Data to a Meaning Representation

Challenges
(with logical inference in general)

- Detecting that
  - Craw is a bird,
  - Bird is an animal
  - Craws can fly but pigs cannot
  - Attending an organization relates to education
  - A person has a mother and a father but can have many children
  - Many more

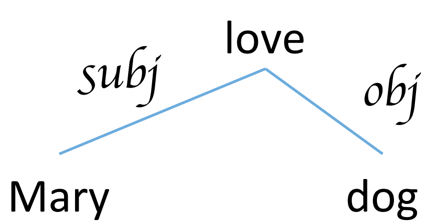
Progress



- Brachman & Levesque, Knowledge representation & reasoning, 2000.
- RTE entailment challenge
  - Since 2005
- Knowledge bases and resources such as Freebase, Wordnet, Yago, dbpedia, …
- Shallow semantic parsers

Mapping to DCS Trees [Tian et al., ACL 2014]

- Dependency-based compositional semantics (DCS) trees [Liang et al., ACL 2011]
  - Similar to (and generated from) dependency parse trees



F1 = love ∩ (Mary[subj] X W[obj])
F2 = animal ∩ π_{obj} (F1)
F3 = have ∩ (John[subj] X F2[obj])

Does John have an animal that Mary love?

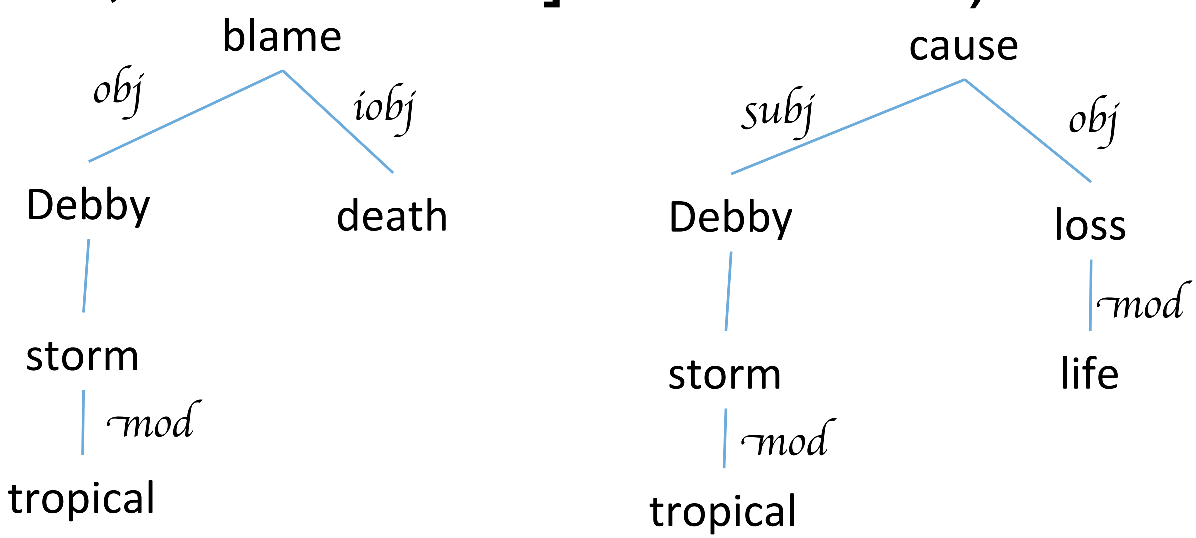
DCS tree node ~ table
Subtree ~ rel. algebra exp.

Logical Inference on DCS

- Some of the axioms
  - $(R \subset S \ \& \ S \subset T) \Rightarrow R \subset T$
  - $R \subset S \Rightarrow \pi_A(R) \subset \pi_A(S)$
  - $W \,!= \varnothing$
- Inference ~ deriving new relations using the tables and the axioms
- Performance on inference problems
  - Comparable to systems in FraCaS and Pascal RTE

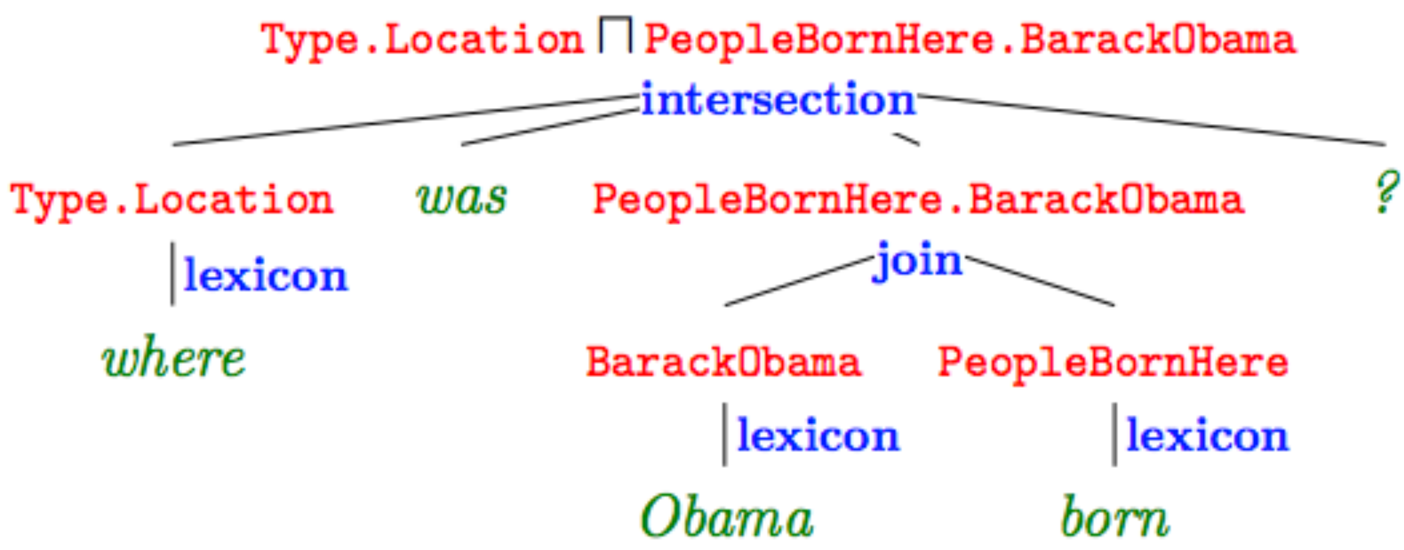
Addressing Knowledge Shortage

- Treat DCS tree fragments as paraphrase candidates
- Establish paraphrases based on distributional similarity (as in [Lewis & Steedman, TACL 2013] and others)



Semantic Parsing using Freebase
[Berant et al., EMNLP 2013]

- Transform questions to freebase derivations
- Learn the mapping from a large collection of question-answer pairs



Approach

- 15 million triplets (text phrases) from ClubWeb09 mapped to Freebase predicates
  - Dates are normalized and text phrases are lemmatized
  - Unary predicates are extracted
    - E.g. city(Chicago) from (Chicago, “is a city in”, Illinois)
    - 6,299 such unary predicates
  - Entity types are checked when there is ambiguity
    - E.g. (BarackObama, 1961) is added to “born in” [person,date] and not to “born in” [person,location]
    - 55,081 typed binary predicates

Two Steps Mapping

- Alignment
  - Map each phrase to a set of logical forms
- Bridging
  - Establish a relation between multiple predicates in a sentence
  - E.g. *Marriage.Spouse.TomCruise* and *2006* will form *Marriage.(Spouse.TomCruise* ∩ *startDate.2006)*

The transformation helps to answer questions using Freebase

Storage and Querying of Triples

- RDF stores
  - Native: Apache Jena TDB, Virtuoso, Algebraix, 4store, GraphDB, …
  - Relational-backed: Jena SDB, C-store, …
- Semantic reasoners
  - Open source: Apache Jena, and many more
  - A list at Manchester U.
    - http://owl.cs.manchester.ac.uk/tools/list-of-reasoners/

Integrating
Natural Language Data

Challenges

- Structure in text
  - Often not known in advance
  - Sometimes subjective
- Optimization and plan generation
  - Difficult with less stats, cost estimates and join dependencies
- Interaction with other systems (e.g. IE, NER)
  - Adds another layer of abstraction

Integration Schemes

- **Tight integration**

- A Rel. Approach to Querying Text
  [Chu et al., VLDB 2007]

- **Lose integration**

- Join queries with external text sources
  [Chaudhuri et al., DIGMOD Record 1995]
- Optimizing SQL queries over text databases
  [Jain et al., ICDE 2008]

A Rel. Approach to Querying Text

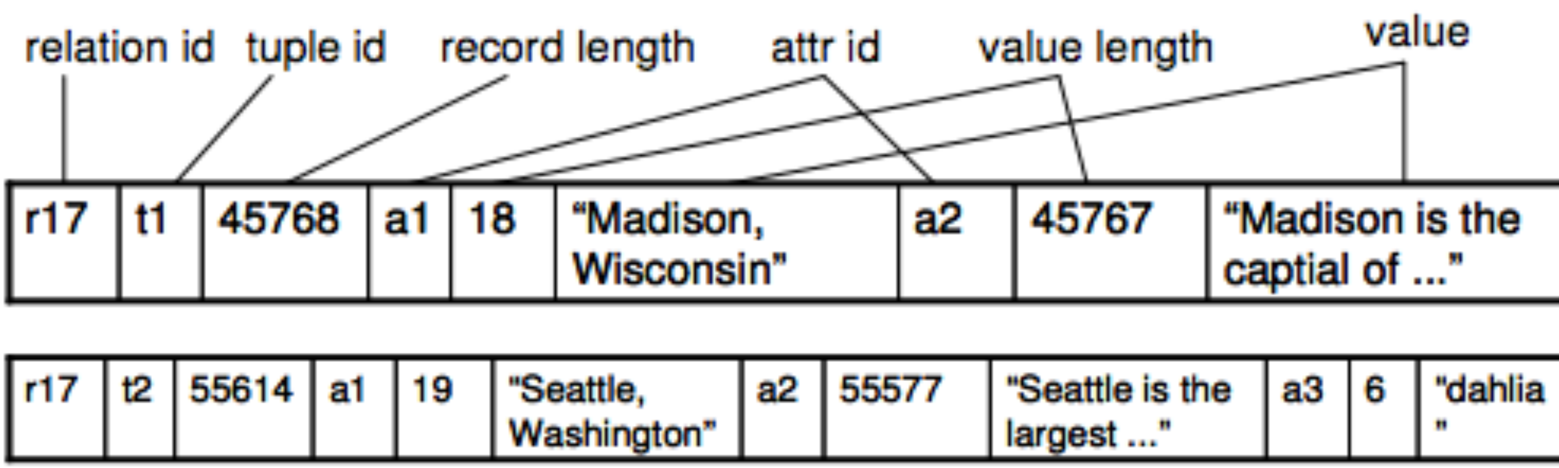
[Chu et al., VLDB 2007]

- Each document is stored in a wide table
- Attributes are added as discovered
- Two tables
  - Attribute catalog
  - Records (one row per document)
- Attributes
  - Two documents can have different attributes
  - Multiple attributes in a doc can have the same name
  - Only non-null values are stored

Attribute Catalog

name	id	type	size
DocTitle	a1	VARCHAR(100)	100
DocContent	a2	TEXT	unlimited
official flower	a3	VARCHAR(50)	50
headquarter.city	a4	VARCHAR(50)	50
headquarter.company	a5	VARCHAR(50)	50

Records



Operators

- Extract
  - Extract desired entities and relationships
- Integrate
  - Suggest mappings between attributes
- Cluster
  - Group documents into one or more clusters

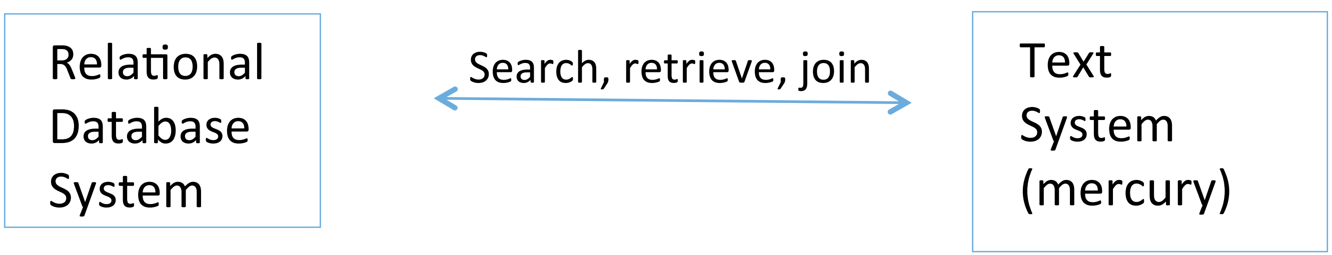
Operator interaction

Integrate(address, sent-to) – extract(city,street,zipcode)

Lose Integration of Text

[Chaudhuri et al., SIGMOD Record 1995]

- Documents stored in a text system
- Relational view of documents



docid	title	author	abstract	...

Integration Techniques

```
SELECT p.member, p.name, m.docid
FROM projects p, mercury m
WHERE p.sponsor='NSF' AND p.name in m.title
        AND p.member in m.author
```

- Tuple substitution
  - Nested loop with the db tuple as the outer relation

Integration Techniques -- Cont.

- Semi-join
  - Suppose the text system can take k terms
  - For n members, send n/k queries of the form (m_1 OR m_2 OR … OR m_k) to the text system
- Probing
  - Select a set of terms (how?) from project title and check their mentions in the text system
  - Keep a list of terms (or assignments) that return empty
- Probing with tuple substitution
  - Maintain a cache

SQL Queries over Text Databases

[Jain et al., ICDE 2008]

- Information Extraction (IE) modules over text
  - headquarter(company, location)
  - ceoOf(company, ceo)
- Relational view of text
  - A set of full outer joins over IE modules
  - e.g. companies =headquarter ⋈ ceoOf ⋈ ...
- SQL queries over relational views
  - Want to improve upon “extract-then-query”

Problem

- Given a SQL query

```
SELECT company, ceo, location
FROM companies
WHERE location='Chicago'
```

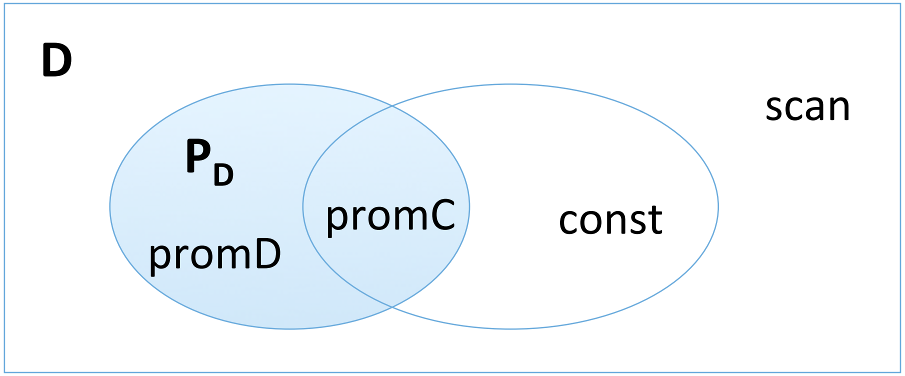
- Find execution strategies that meet some efficiency and quality constraints
  - In terms of runtime, precision, recall, …
- On-the-fly IE from text

Retrieval Strategies

- scan
  - Process all documents
- const chicago
  - Process documents that contain query keywords
- promD headquarter OR (based AND shares)
  - Only process the promising documents for each IE system (using IE specific keywords)
- promC chicago AND (Headquarter OR (based AND shares)
  - AND the predicates of const and promD

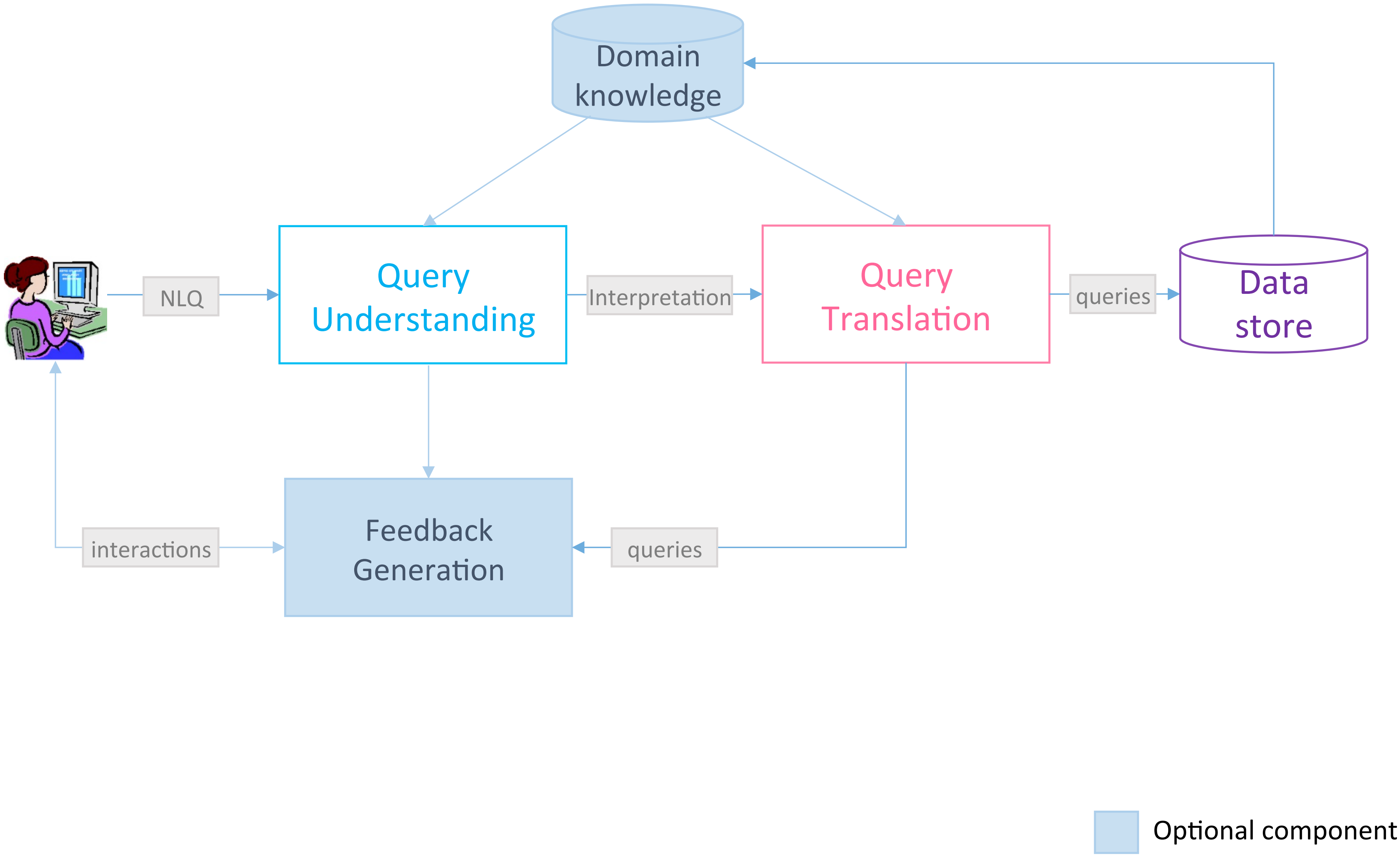
Selecting an Execution Plan

- Stats estimated for each strategy
  - # of matching docs   docs(E, promC, D)
  - Retrieval time   rTime(E, scan, D)
- Cost estimation
  - Stratified sampling (with one stratum for P_D and another stratum for $D-P_D$)
  - For const use both strata
  - For promC & promD use P_D only



Natural Language Interface to Databases (NLIDB)

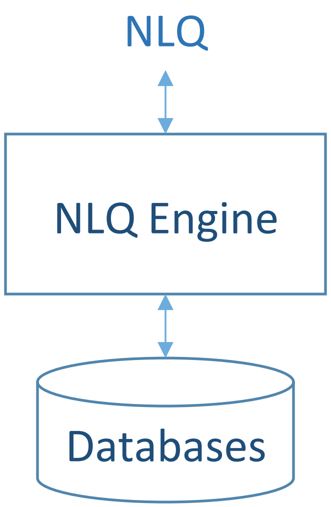
Anatomy of a NLIDB



Query Understanding
– Scope of Natural Language Support

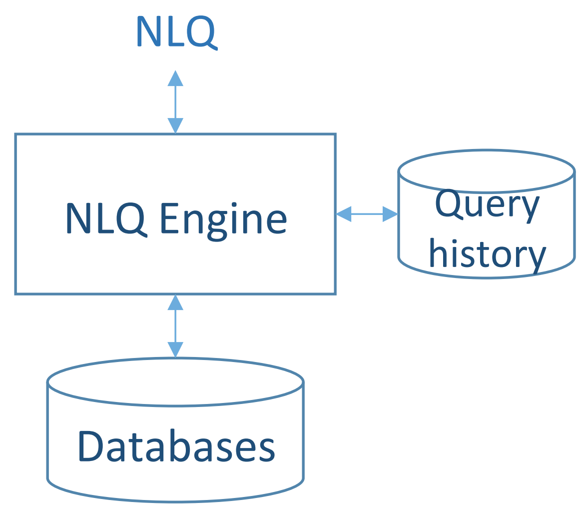
Query Understanding – Stateless and Stateful

Stateless NLQs



Each query must be
- Fully specified
- Processed independently

Stateful NLQs



Each query
- Can be partially specified
- Processed with regards to previous queries

Query Understanding - Parser Error Handling

Parsers make mistakes.

- **News**: Accuracy of a dependency parser = ~90% [Andor et al., 2016]
- **Questions**: ~80% [Judge et al., 2006]

Different approaches:

Ignore

- Do nothing

Auto-correction

- Detect and correct certain parser mistakes

Interactive correction

- Query reformulation
- Parse tree correction

Query Translation - Bridging the Semantic Gaps

- **Vocabulary gap**

  "*Bill Clinton*" vs. "*William Jefferson Clinton*"

  "*IBM*" vs. "*International Business Machine Incorporated*"

- **Leaky abstraction**
  - Mismatch between abstraction (e.g. data schema/domain ontology) and user assumptions

    "*top executives*" vs "*person with title CEO, CFO, CIO, etc.*"

- **Ambiguity in user queries**
  - Underspecified queries

    "*Watson movie*" → "*Watson"* as actor/actress

    E.g. Emma Watson

    "*Watson*" as a movie character

    E.g. *Dr. Watson* in movie "Holmes and Watson"

    …

Query Translation – Query Construction

- **Approaches**
  - Machine learning
  - Construct formal queries from NLQ interpretations with deterministic algorithms

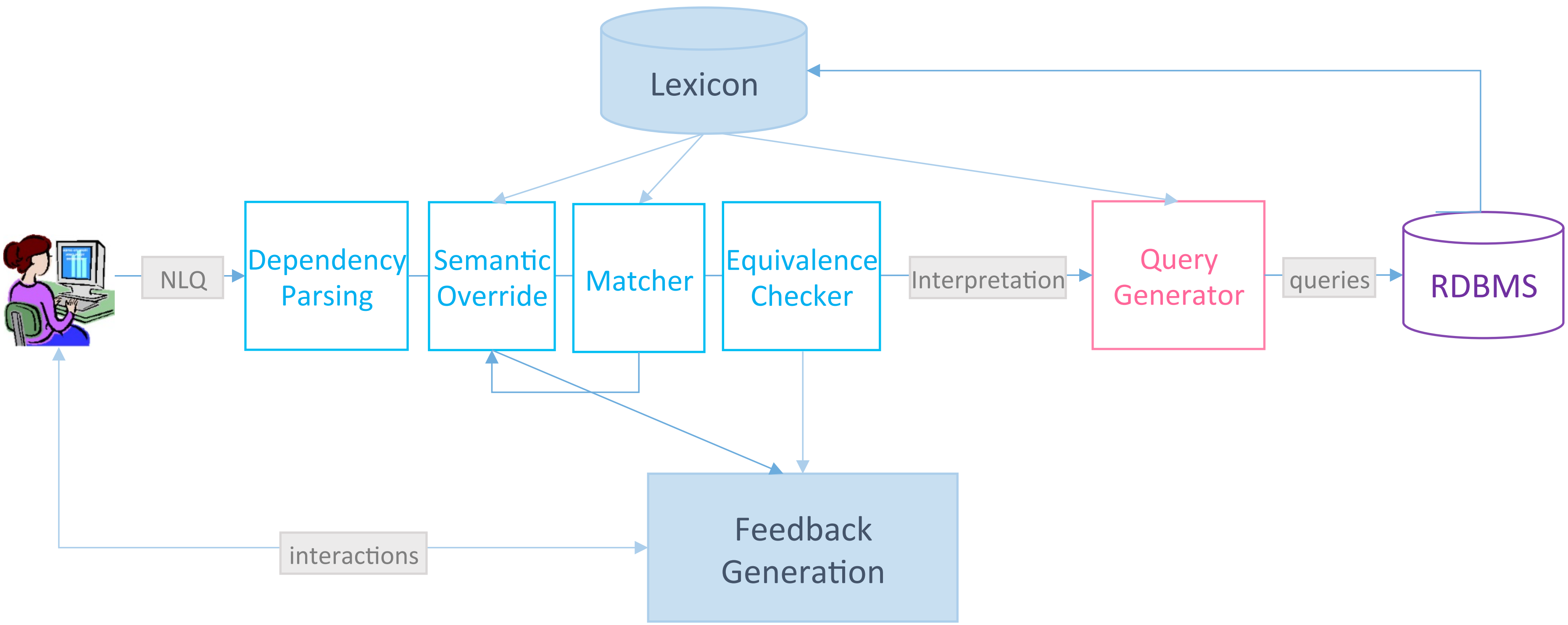
- **Query**
  - Formal query languages (e.g. XQuery / SQL)
  - Intermediate language independent of underlying data stores
    - The same intermediate query for different data stores

Systems

- PRECISE
- NaLIX
- NLPQC
- FREyA
- NaLIR
- ML2SQL
- NL_2CM
- ATHANA

PRECISE [Popescu et al., 2003,2004]

- Controlled NLQ based on Semantic Tractability



PRECISE [Popescu et al., 2003,2004]

- **Semantic Tractability**

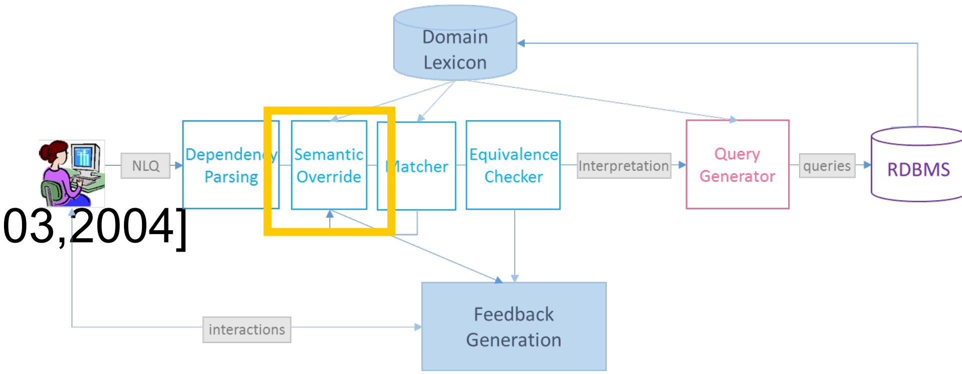
  Database element: relations, attributes, or values

  Token: a set of word stems that matches a database element

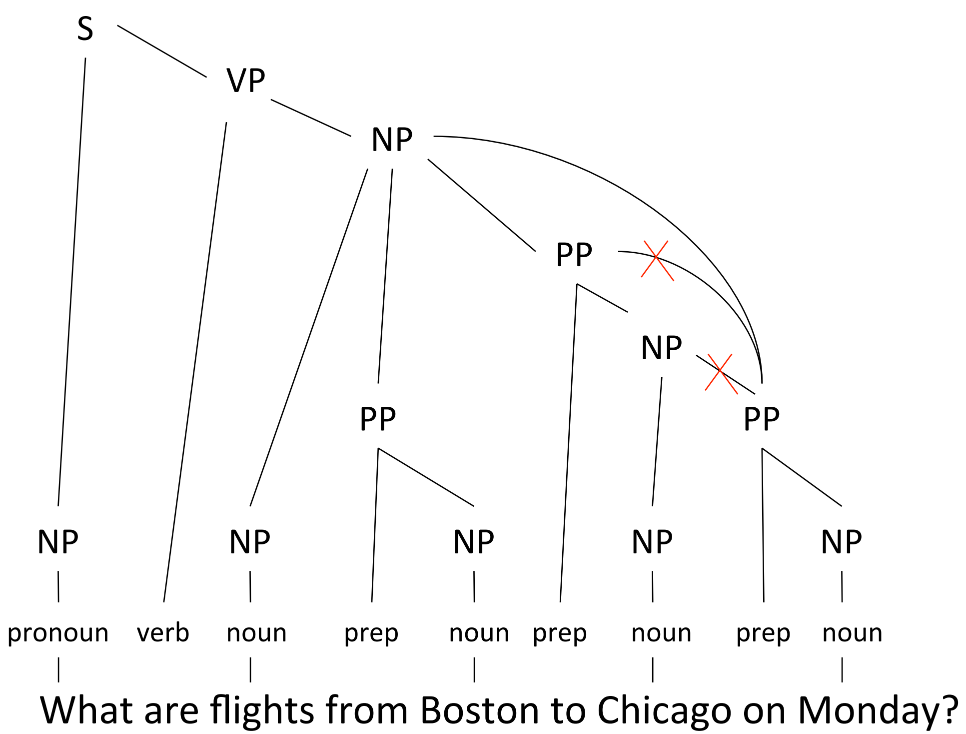
  Syntactic marker: a term from a fixed set of database-independent terms that make no semantic contribution to the interpretation of the NLQ

  Semantically tractable sentence: Given a set of database element *E*, a sentence S is considered semantic tractable, when its complete tokenization satisfies the following conditions:
    - Every token matches a unique data element in E
    - Every attribute token attaches to a unique value token
    - Every relation token attaches to either an attribute token or a value token

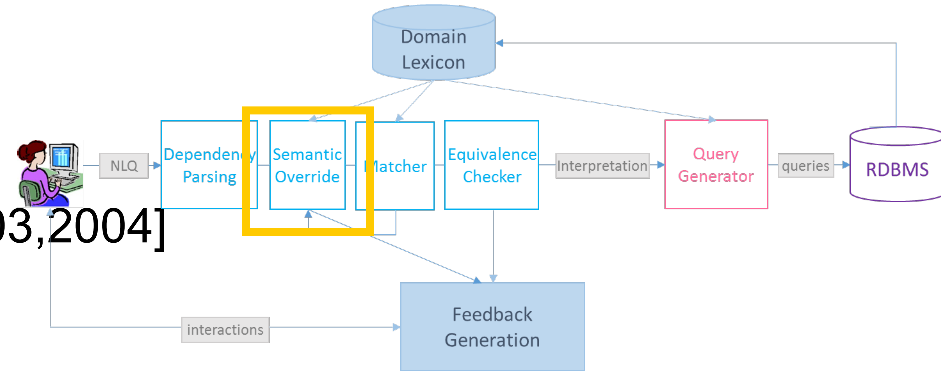
PRECISE [Popescu et al., 2003,2004]



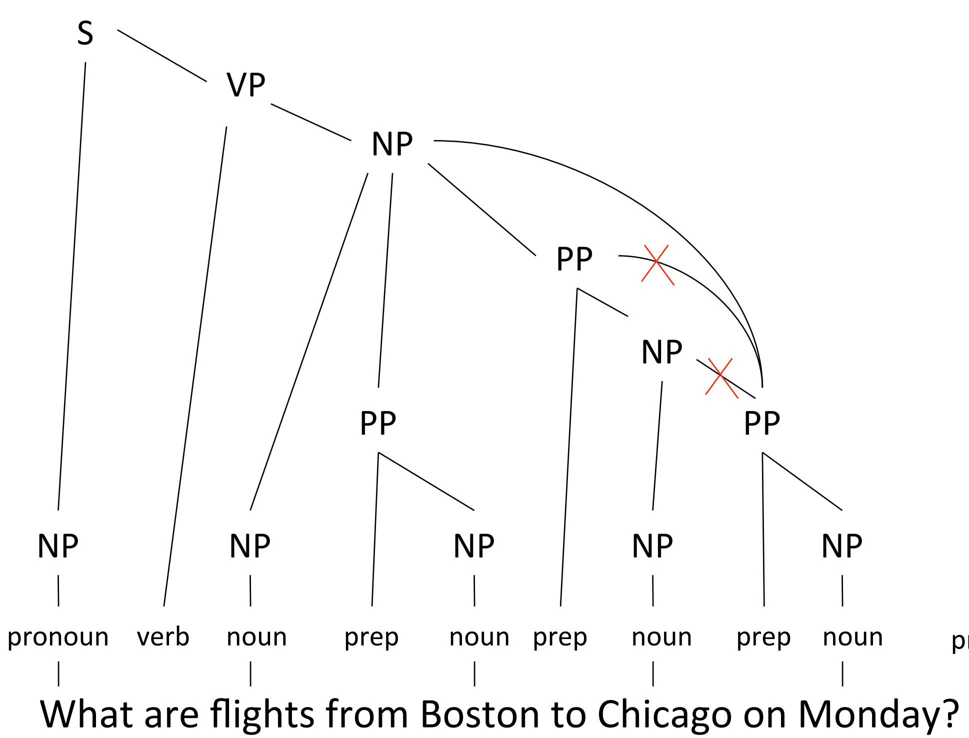
- Explicitly correct parsing errors:
  - Preposition attachment
  - Preposition ellipsis

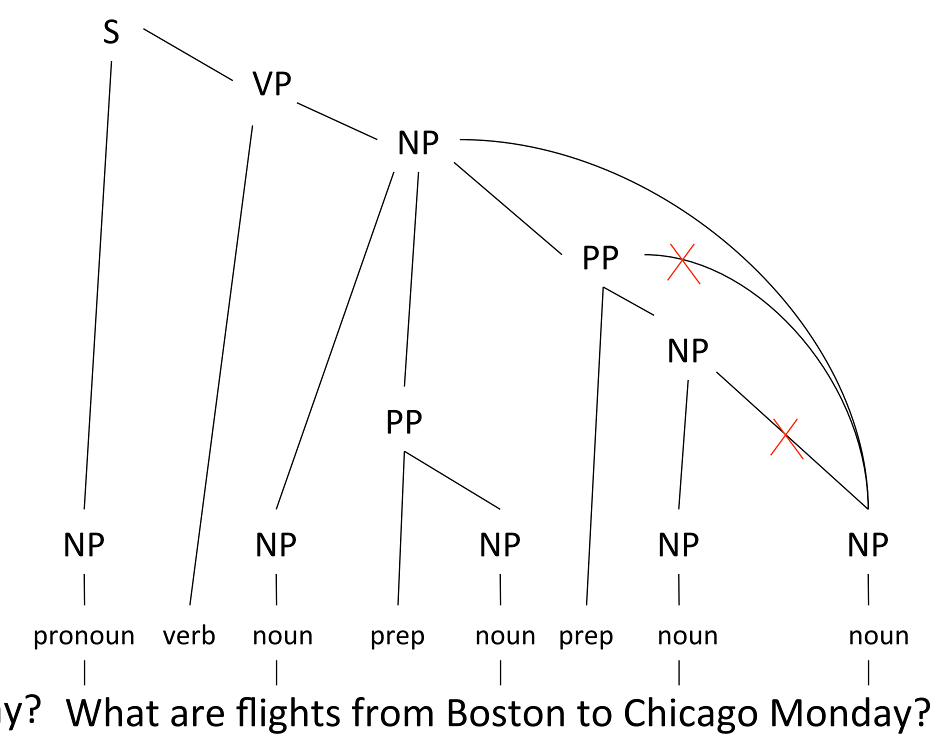


PRECISE [Popescu et al., 2003,2004]

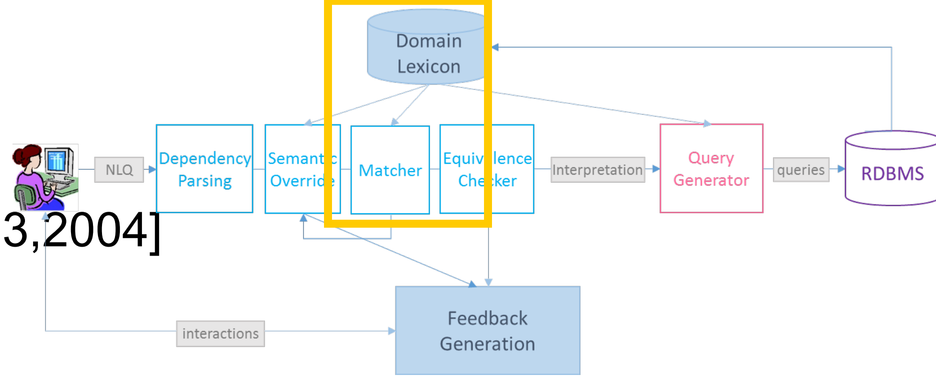


- Explicitly correct parsing errors:
  - Preposition attachment
  - Preposition ellipsis

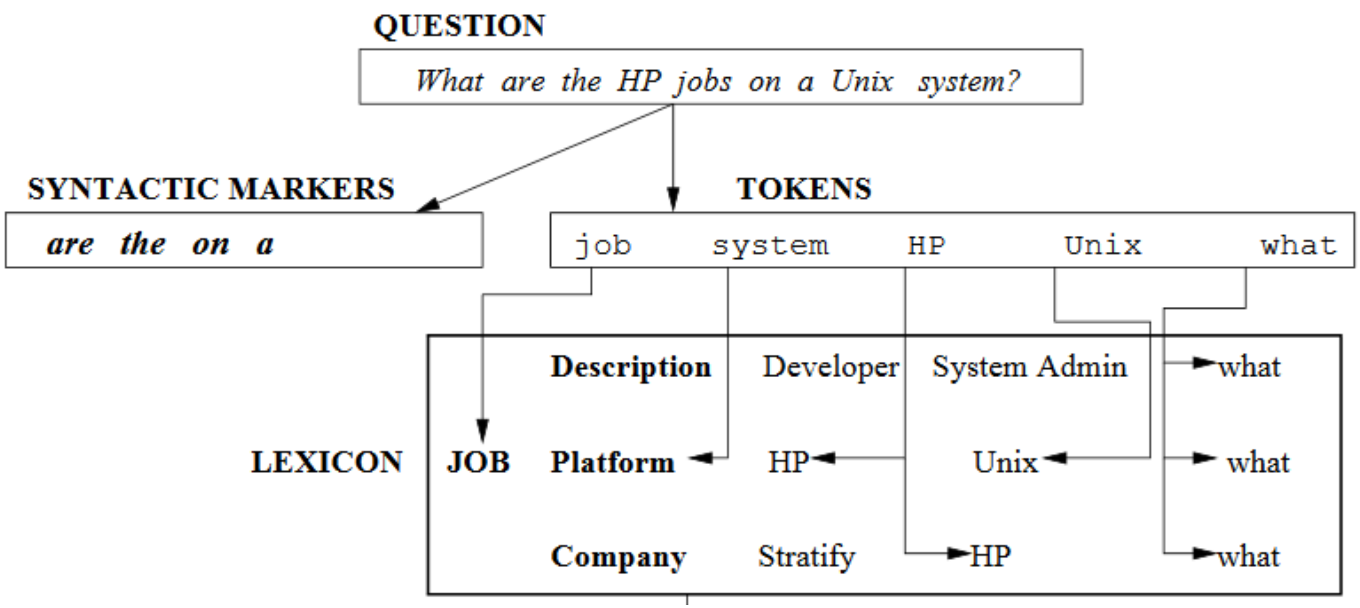




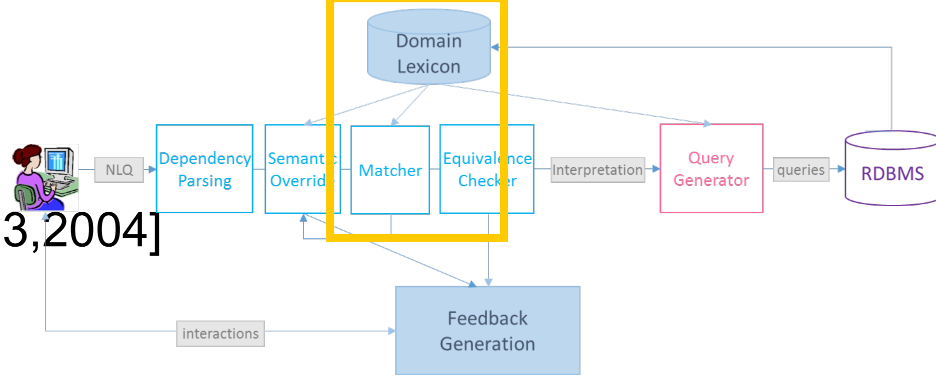
PRECISE [Popescu et al., 2003,2004]



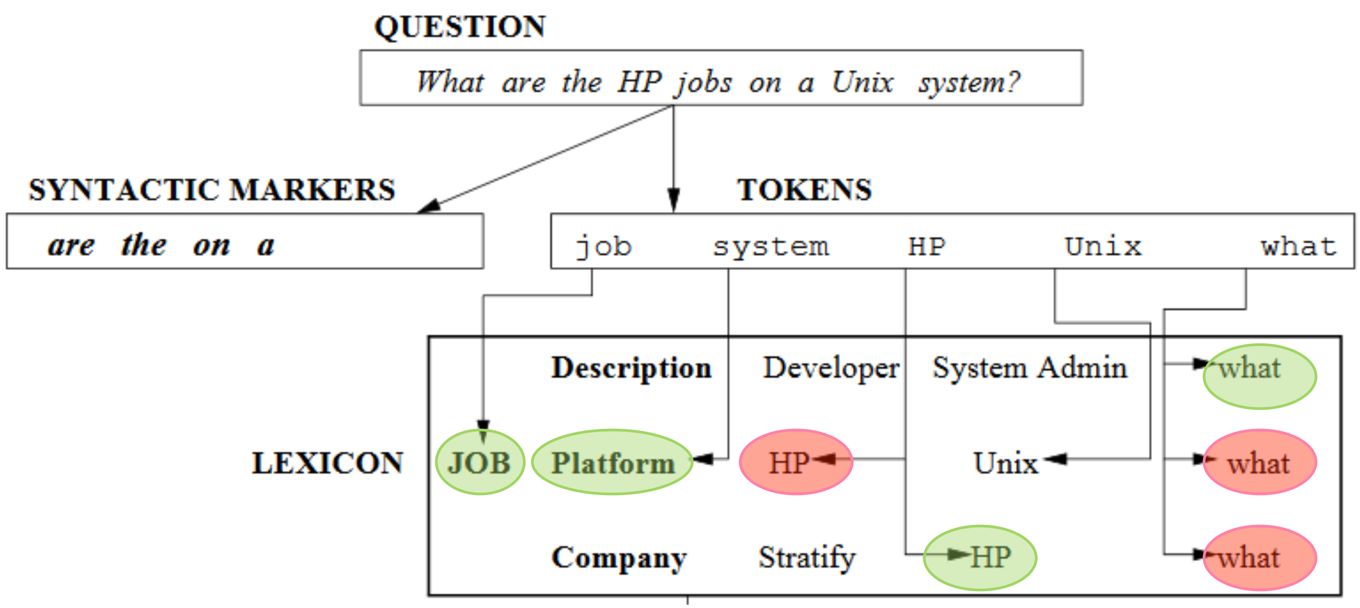
- Mapping parse tree nodes based on lexicon built from database



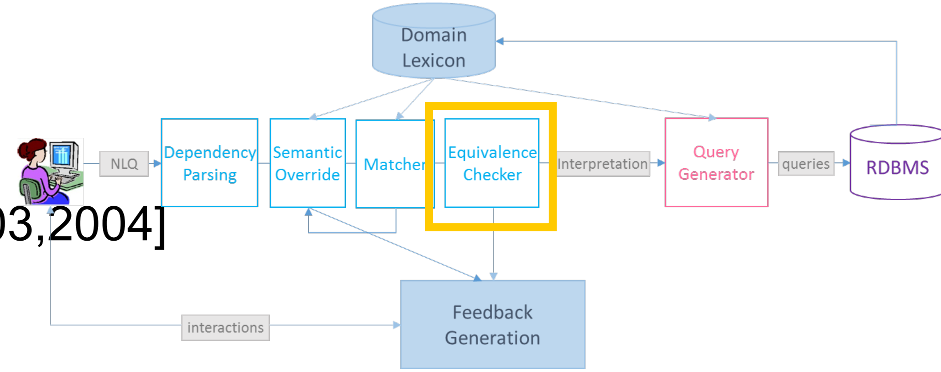
PRECISE [Popescu et al., 2003,2004]



- Addressing ambiguities through lexicon + semantic tractability
  - Maximum-flow solution



PRECISE [Popescu et al., 2003,2004]



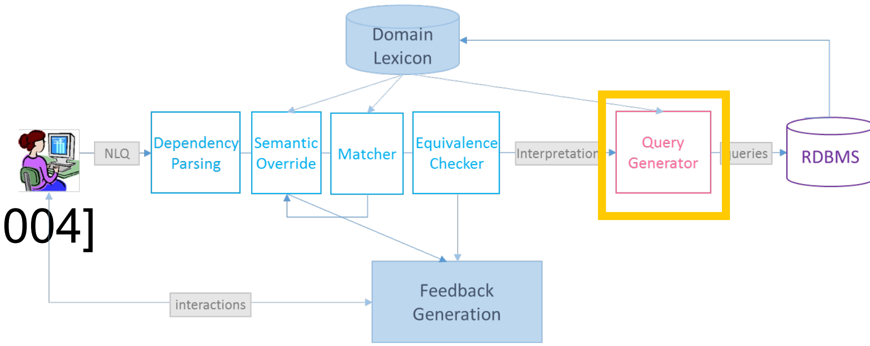
- Addressing ambiguities through lexicon + semantic tractability + user input

NLQ

What are the systems analyst jobs in Austin?

Interpretation 1 **Job title**: systems analyst

Interpretation 2 **Area**: systems
Job title: analyst



PRECISE [Popescu et al., 2003,2004]

- 1-to-many translation from interpretations to SQL based on all possible join-paths

NLQ

What are the HP jobs on Unix in a small town?

Interpretations

Job.Description ← What
Job.Company ← *'HP'*
Job.Platform ← *'Unix'*
City.size ← 'small'

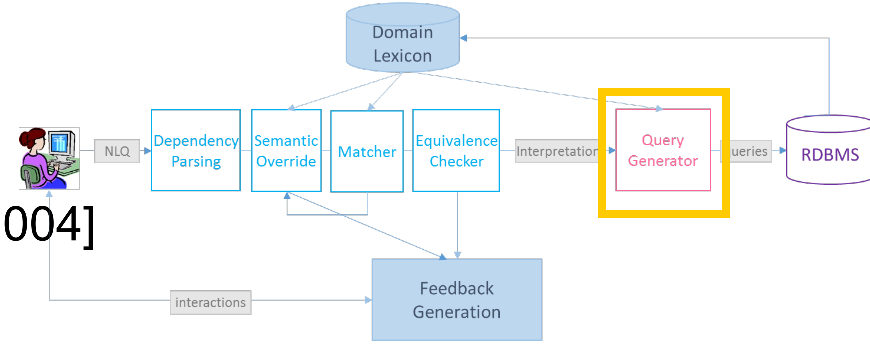
DB Schema

Job
JobID
Description
Company
Platform

City
CityID
Name
State
Size

```
SELECT DISTINCT Job.Description
FROM Job, City
WHERE Job.Platform = 'HP'
  AND Job.Company = 'Unix'
  AND Job.JobID = City.CityID
```

PRECISE [Popescu et al., 2003,2004]



- 1-to-many translation from interpretations to SQL based on all possible join-paths

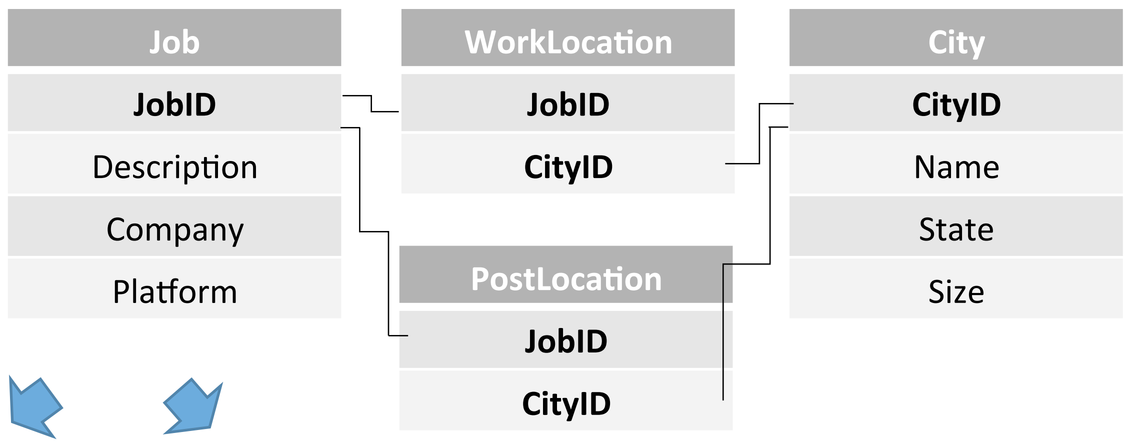
NLQ

What are the HP jobs on Unix in a small town?

Interpretations

Job.Description ← What
Job.Company ← *'HP'*
Job.Platform ← *'Unix'*
City.size ← 'small'

DB Schema

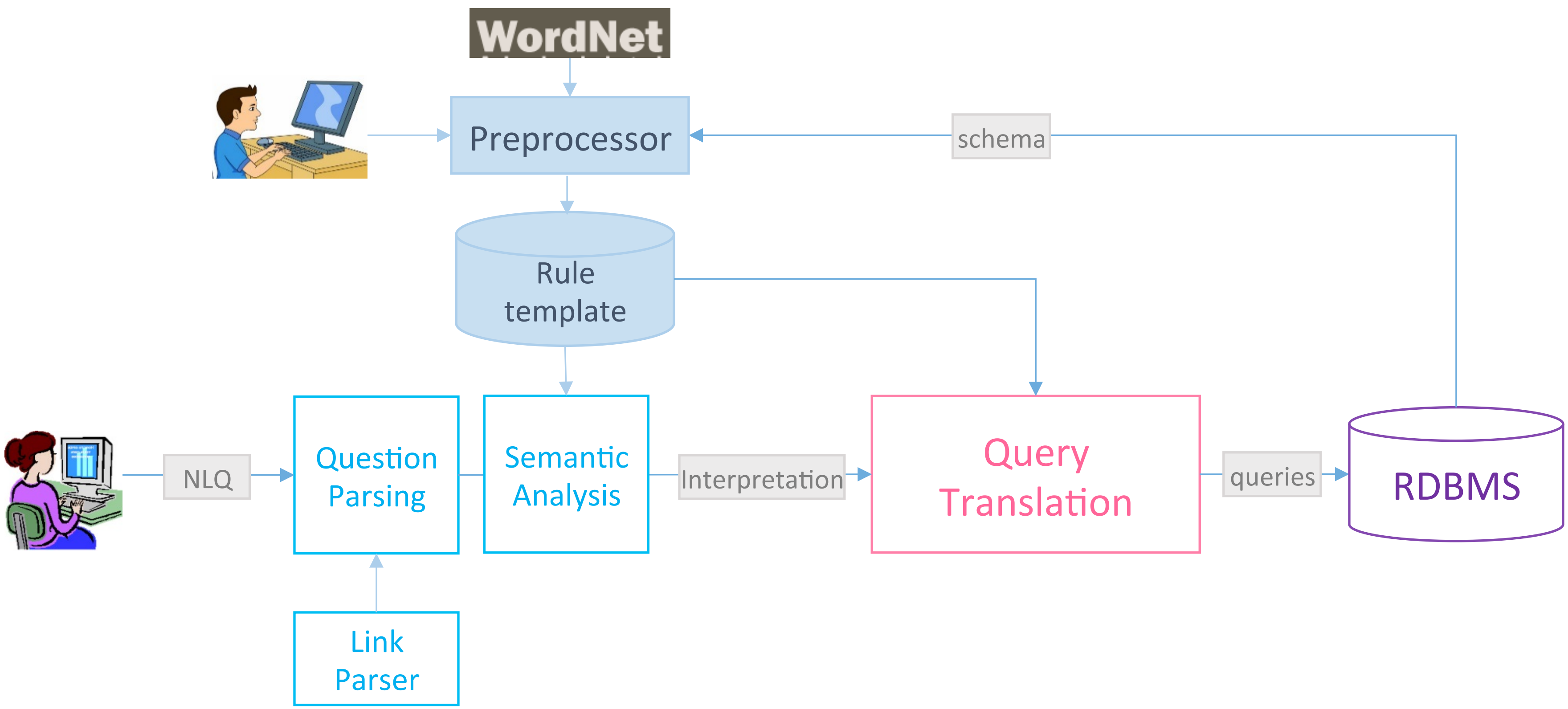



```
SELECT DISTINCT Job.Description
FROM Job, WorkLocation, City
WHERE Job.Platform = 'HP'
  AND Job.Company = 'Unix'
  AND Job.JobID = WorkLocation.JobID
  AND WorkLocation.CityID = City.CityID
```

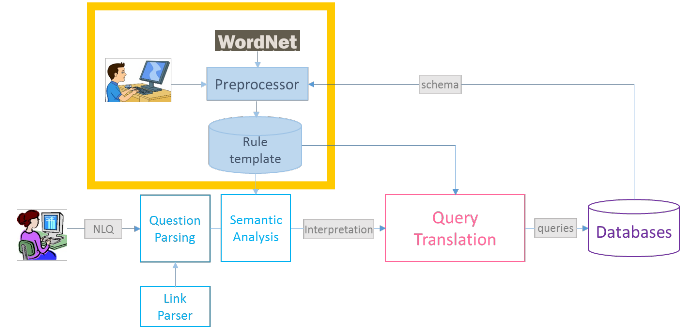
```
SELECT DISTINCT Job.Description
FROM Job, PostLocation, City
WHERE Job.Platform = 'HP'
  AND Job.Company = 'Unix'
  AND Job.JobID = WorkLocation.JobID
  AND PostLocation.CityID = City.CityID
```

NLPQC [Stratica et al., 2005]

- Controlled NLQ based on predefined rule templates
- No query history

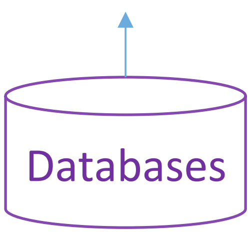


NLPQC [Stratica et al., 2005]



- Build mapping rules for table names and attributes
  - Automatically generated using WordNet
  - Curated by system administrator

Table name: *resource*
…



WordNet

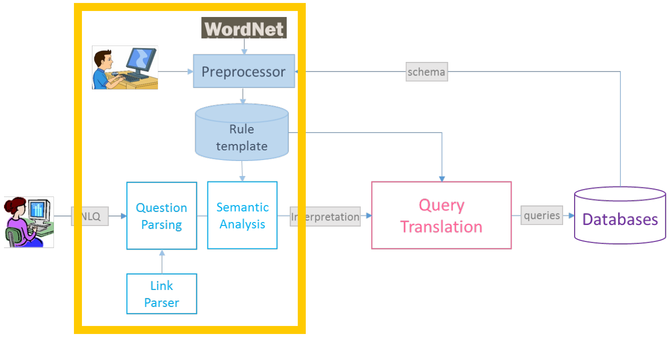
Synonyms: 3 sense of resource
Sense 1: resource
Sense 2: resource
Sense 3: resource, resourcefulness, imagination
Hypernyms: 3 sense of resource
…
Hyponyms: 3 sense of resource
…

accept/reject/add

Semantic set name	Elements
resource	resource, book, volume, record, script
resource.title	title, name, rubric, caption, legend
resource.language	language, speech, words, source language
resource.keyword	keyword, key word

…

NLPQC [Stratica et al., 2005]



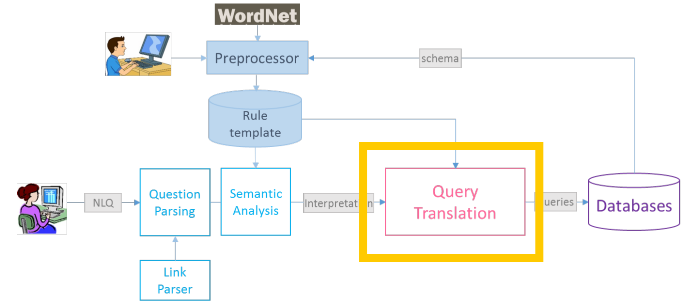
- Mapping parse tree node to data schema and value based on mapping rules

NLQ

Who is the author of book Algorithms

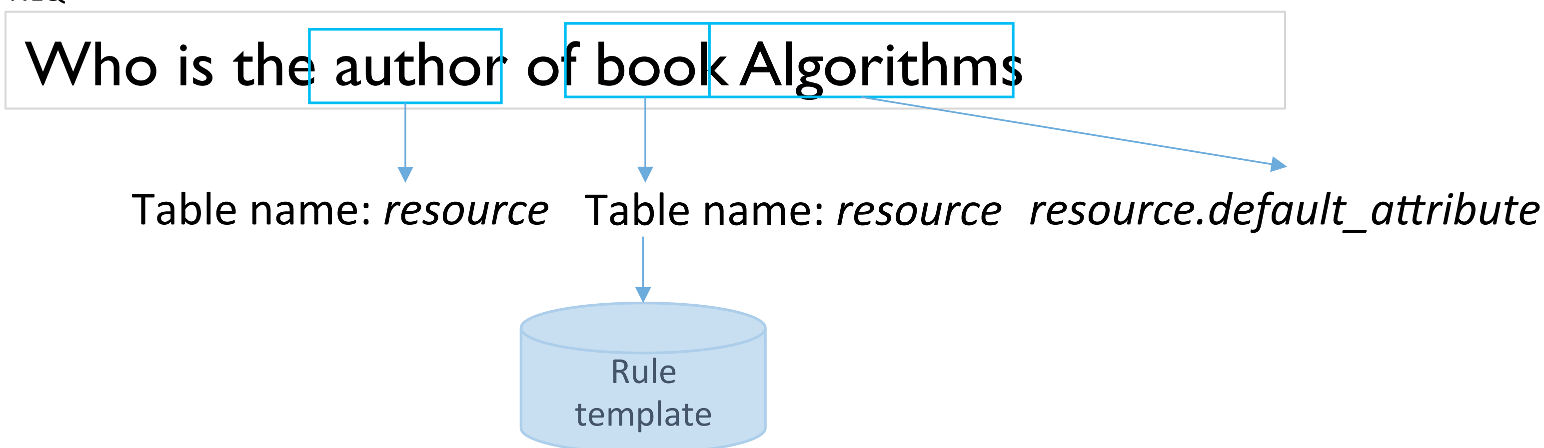
Table name: *resource* Table name: *resource* *resource.default_attribute*

NLPQC [Stratica et al., 2005]



- Mapping parse tree node to data schema and value based on pre-defined mapping rules
- Mapping parse trees to SQL statements based on pre-defined rule templates

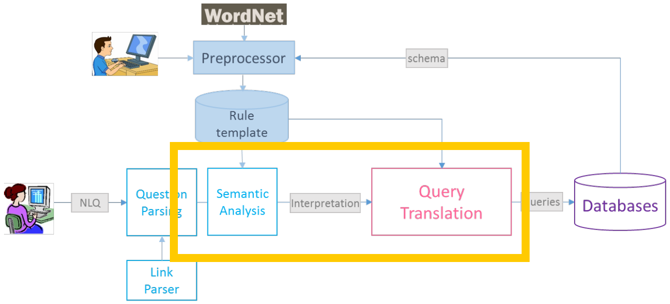
NLQ




```
if (table author and table resource are used) then
      (related table resource_author is used too) and
      (SQL query template includes
            resource_author.resource_id=resource.resource_id
            AND resource_author.author_id=author.author_id)
```

```
SELECT author.name FROM author, resource, resource_author
WHERE  resource.title = "Algorithm"
AND    resource_author.resource_id=resource.resource_id
AND    resource_author.author_id=author.author_id
```

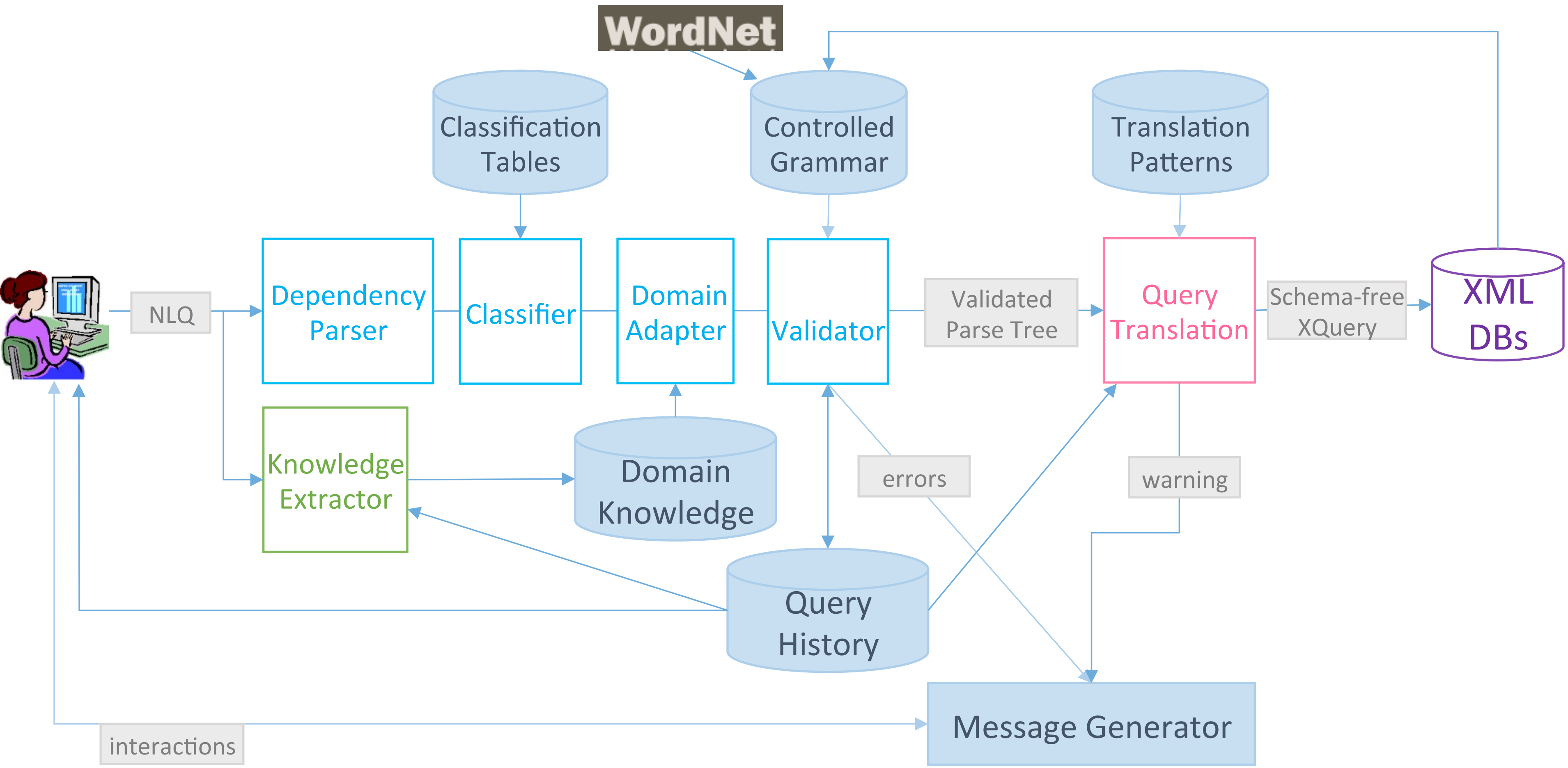
NLPQC [Stratica et al., 2005]



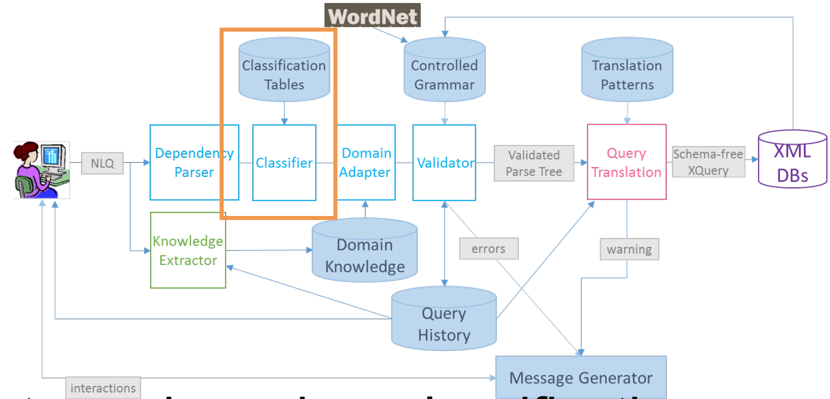
- No explicit ambiguity handling → leave it to mapping rules and rule templates
- No parsing error handling → Assume no parsing error

NaLIX [Li et al., 2007a, 2007b, 2007c]

• Controlled NLQ based on pre-defined controlled grammar



NaLIX [Li et al., 2007a, 2007b, 2007c]



- Classify parse tree nodes into different types based on classification tables
  - **Token**: words/phrases that can be mapped into a XQuery component
    - Constructs in FLOWR expressions
  - **Marker**: word/phrase that cannot be mapped into a XQuery component
    - Connecting tokens, modify tokens, pronoun, stopwords

NLQ

What are the state that share a watershed with California

Classified parse tree

What are [CMT]

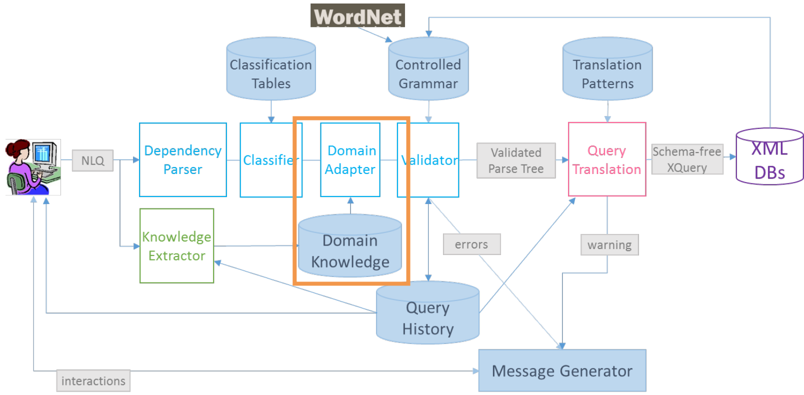
state[NT]

the [MM] share [CM]

the [MM] watershed [NT] with [CM]

a [MM] California [VT]

NaLIX [Li et al., 2007a, 2007b, 2007c]

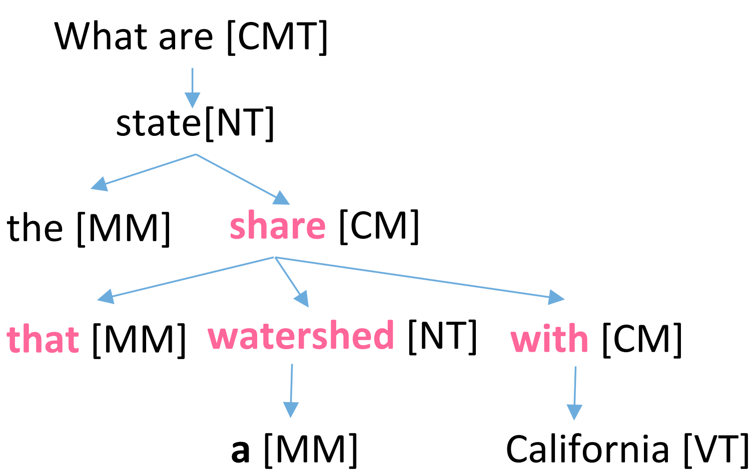


- Expand scope of NLQ support via domain adaptation

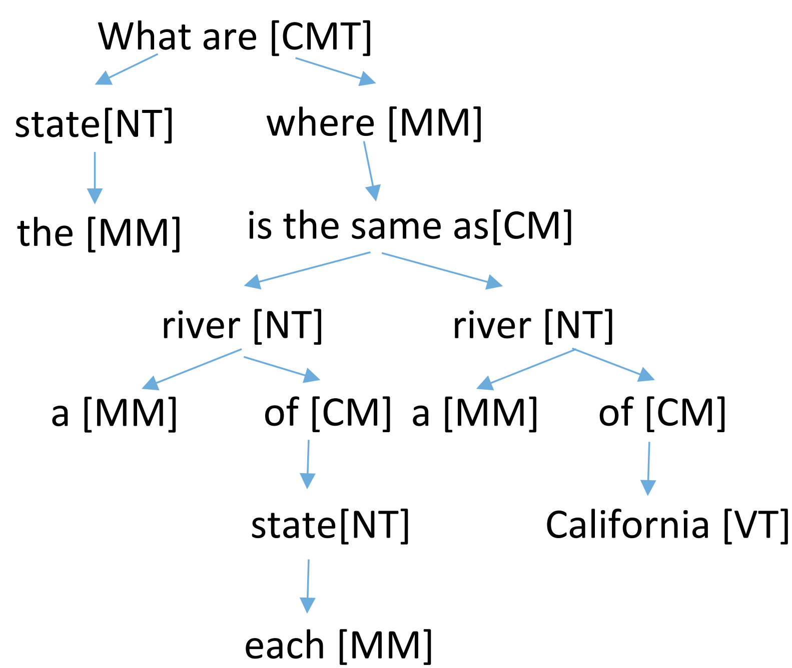
NLQ

What are the state that share a watershed with California

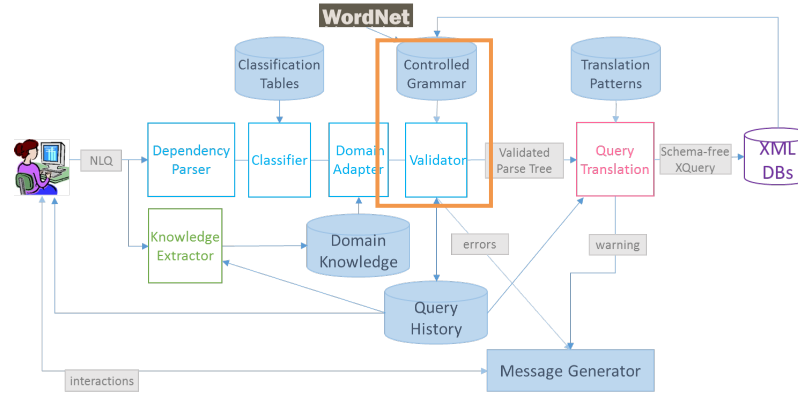
Classified parse tree



Updated classified parse tree with domain knowledge



NaLIX [Li et al., 2007a, 2007b, 2007c]



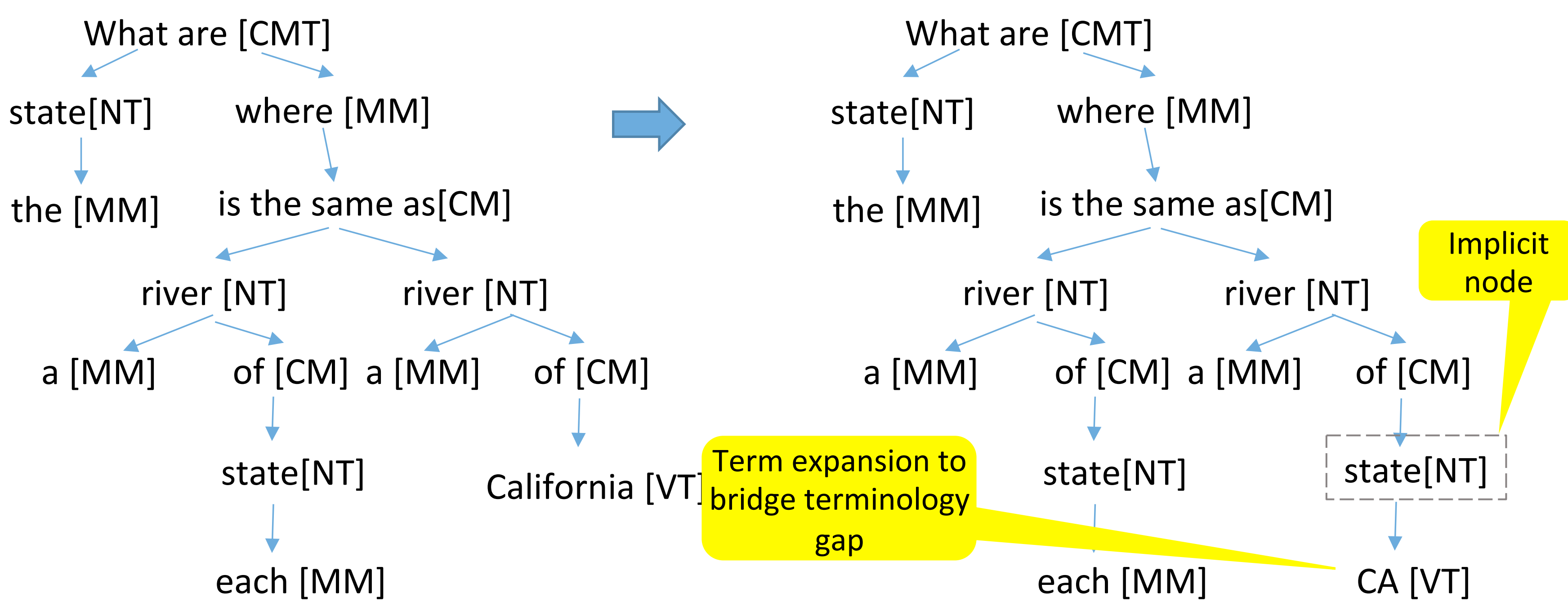
- Validate classified parse tree + term expansion + insert implicit nodes

NLQ

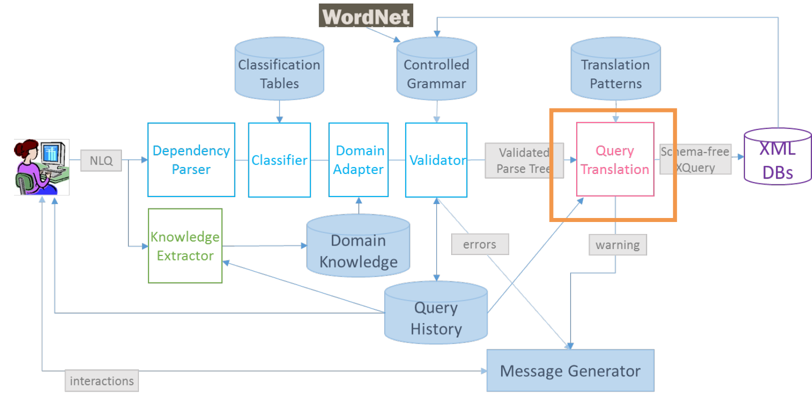
What are the state that share a watershed with California

Updated classified parse tree with domain knowledge

Updated classified parse tree post validation



NaLIX [Li et al., 2007a, 2007b, 2007c]

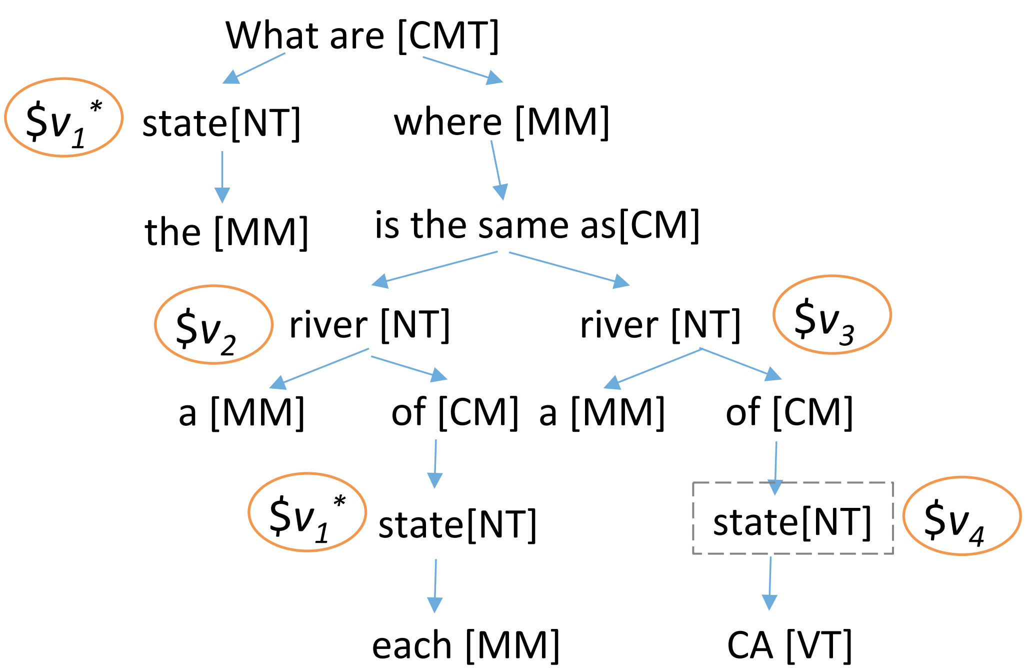


- Translation: (1) Variable binding

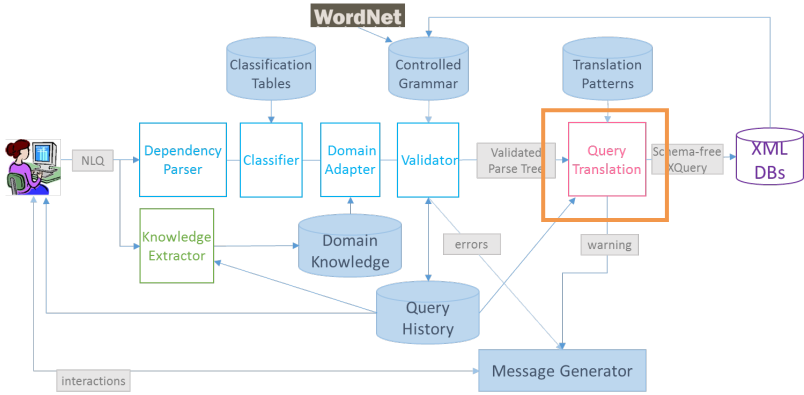
NLQ

What are the state that share a watershed with California

Updated classified parse tree post validation



NaLIX [Li et al., 2007a, 2007b, 2007c]

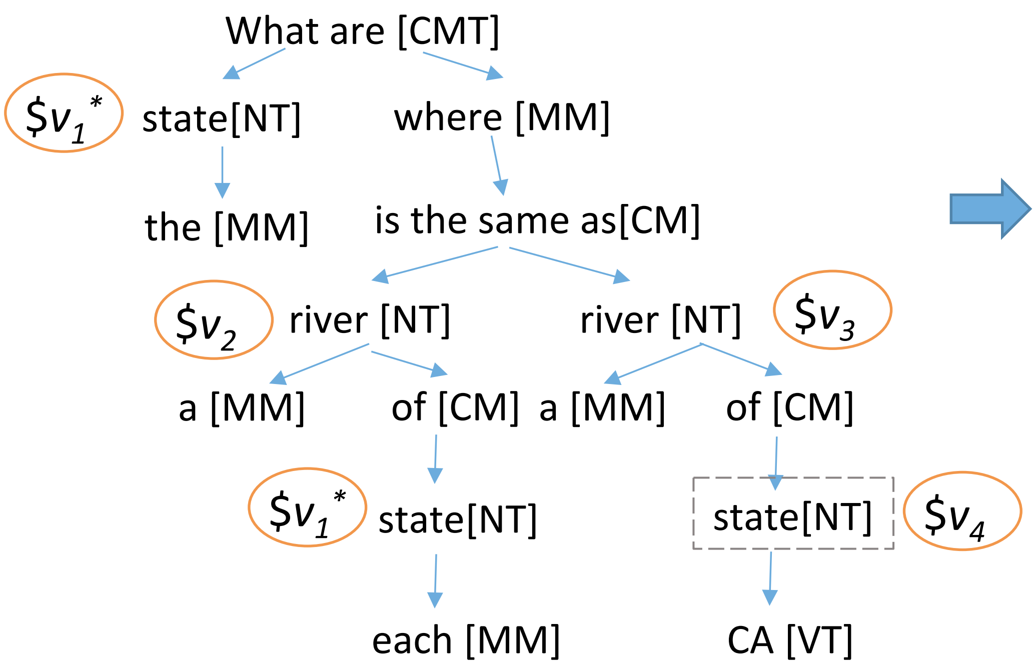


- Translation: (2) Pattern Mapping

NLQ

What are the state that share a watershed with California

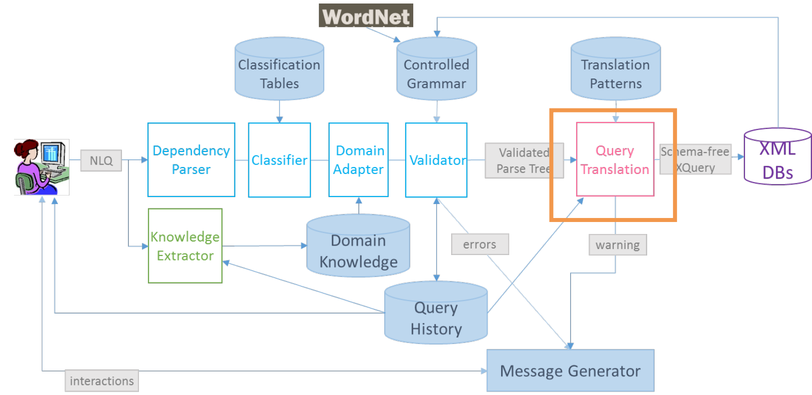
Updated classified parse tree post validation



XQuery fragments

```
for $v1 in ⟨doc⟩//state
for $v2 in ⟨doc⟩//river
for $v3 in ⟨doc⟩//river
for $v4 in ⟨doc⟩//state
where $v2 = $v3
where $v4 = "CA"
```

NaLIX [Li et al., 2007a, 2007b, 2007c]

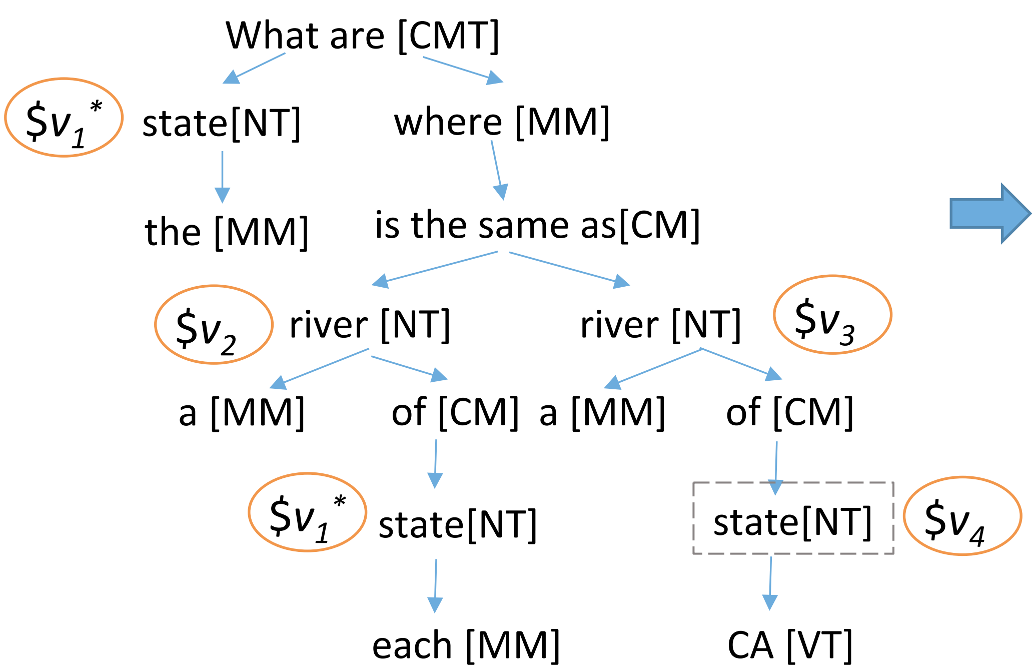


- Translation: (3) Nesting and grouping

NLQ

What are the state that share a watershed with California

Updated classified parse tree post validation

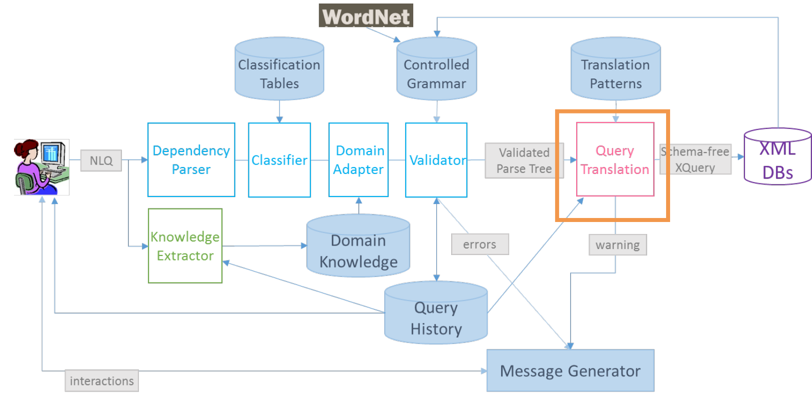


XQuery fragments

```
for $v1 in ⟨doc⟩//state
for $v2 in ⟨doc⟩//river
for $v3 in ⟨doc⟩//river
for $v4 in ⟨doc⟩//state
where $v2 = $v3
where $v4 = "CA"
```

No aggregation function/qualifier
→ No nesting/grouping

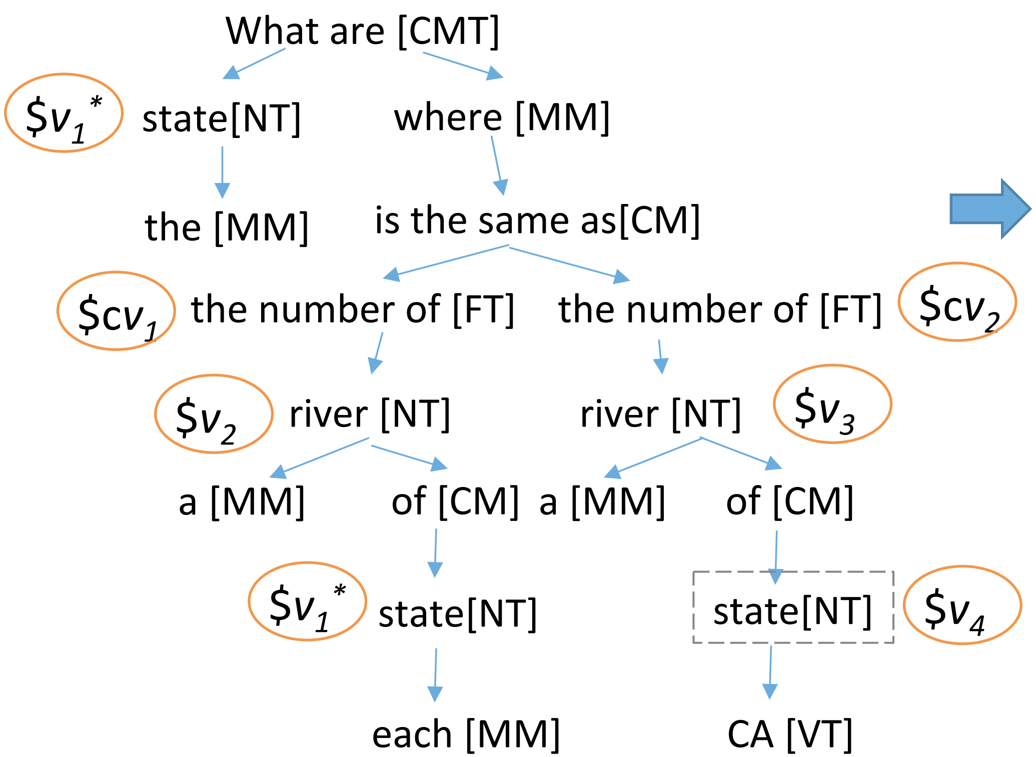
NaLIX [Li et al., 2007a, 2007b, 2007c]



- Translation: (3) Nesting and grouping

NLQ

Find all the states whose number of rivers is the same as the number of rivers in California?

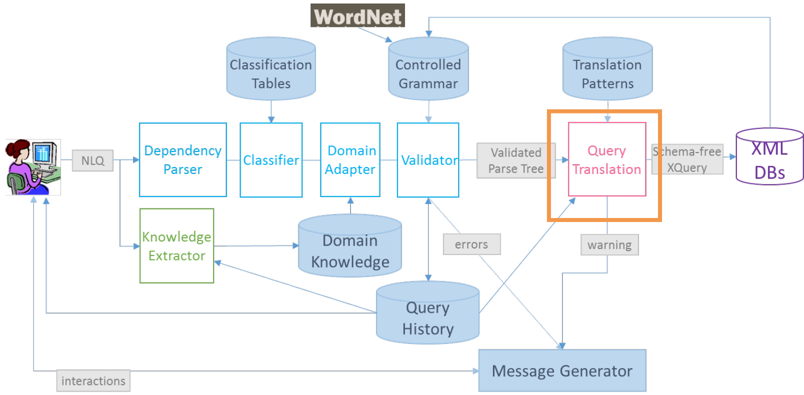


XQuery fragments

<u>for</u> $\$v_1$ <u>in</u> ⟨*doc*⟩//state
<u>for</u> $\$v_2$ <u>in</u> ⟨*doc*⟩//river
<u>for</u> $\$v_3$ <u>in</u> ⟨*doc*⟩//river
<u>for</u> $\$v_4$ <u>in</u> ⟨*doc*⟩//state
<u>for</u> $\$cv_1$ = <u>count</u>($\$v_2$)
<u>for</u> $\$cv_2$ = <u>count</u>($\$v_3$)
<u>where</u> $\$cv_1 = \$cv_2$
<u>where</u> $\$v_4$ = "CA"

Aggregation function
→ Nesting and grouping based on $\$v_2$ and $\$v_3$

NaLIX [Li et al., 2007a, 2007b, 2007c]



- Translation: (4) Construction full query

NLQ

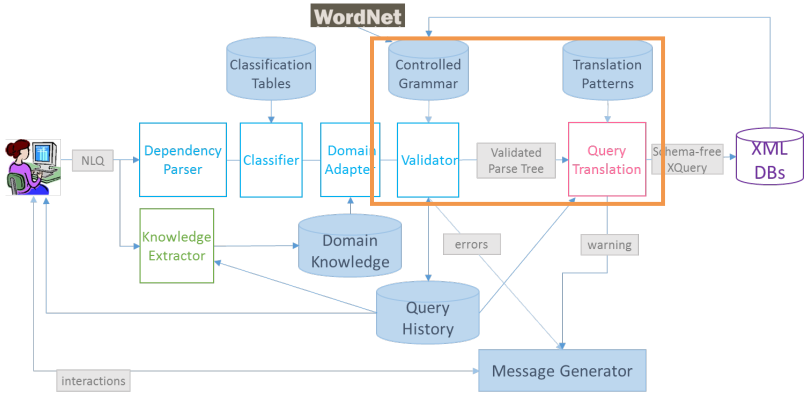
Find all the states whose number of rivers is the same as the number of rivers in California?

XQuery fragments

```
for $v1 in ⟨doc⟩//state
for $v2 in ⟨doc⟩//river
for $v3 in ⟨doc⟩//river
for $v4 in ⟨doc⟩//state
for $cv1 = count($v2)
for $cv2 = count($v3)
where $cv1 = $cv2
where $v4 = "CA"
```

```
for $v1 in doc("geo.xml")//state,
    $v4 in doc("geo.xml")//state
let $vars1 := {
    for $v2 in doc("geo.xml")//river,
        $v5 in doc("geo.xml")//state
    where mqf($v2,$v5)
      and $v5 = $v1
    return $v2}
let $vars2 := {
    for $v3 in doc("geo.xml")//river,
        $v6 in doc("geo.xml")//state
    where mqf($v3,$v6)
      and $v6 = $v4
    return $v3}
where count($vars1) = count($vars2)
  and $v4 = "CA"
return $v1
```

NaLIX [Li et al., 2007a, 2007b, 2007c]



- Support partially specified follow-up queries
- Detect topic switch to refresh query context

NLQ

How about with Texas?

Substitution marker

Validated parse tree

How about [SM]

with [CM]

TX [VT]

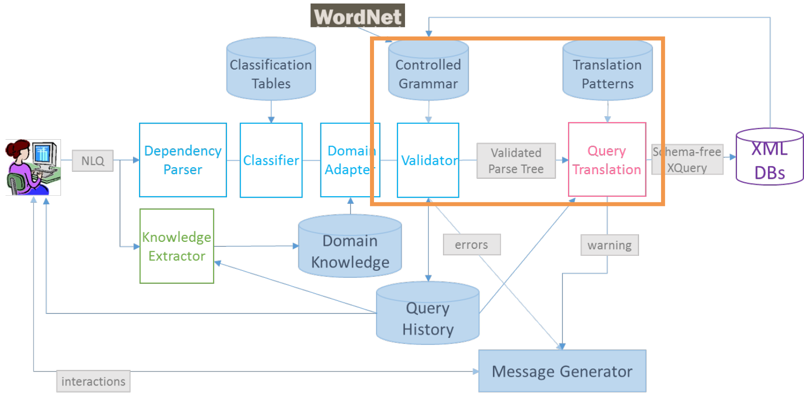
Query context

```
for $v1 in ⟨doc⟩//state
for $v2 in ⟨doc⟩//river
for $v3 in ⟨doc⟩//river
for $v4 in ⟨doc⟩//state
for $cv1 = count($v2)
for $cv2 = count($v3)
where $cv1 = $cv2
where $v4 = "CA"
```

Updated query context

```
for $v1 in ⟨doc⟩//state
for $v2 in ⟨doc⟩//river
for $v3 in ⟨doc⟩//river
for $v4 in ⟨doc⟩//state
for $cv1 = count($v2)
for $cv2 = count($v3)
where $cv1 = $cv2
where $v4 = "TX"
```

Updated value



NaLIX [Li et al., 2007a, 2007b, 2007c]

- Handle ambiguity
  - Ambiguity in terms → User feedback

    e.g. "California" can be the name of a state, as well as a city

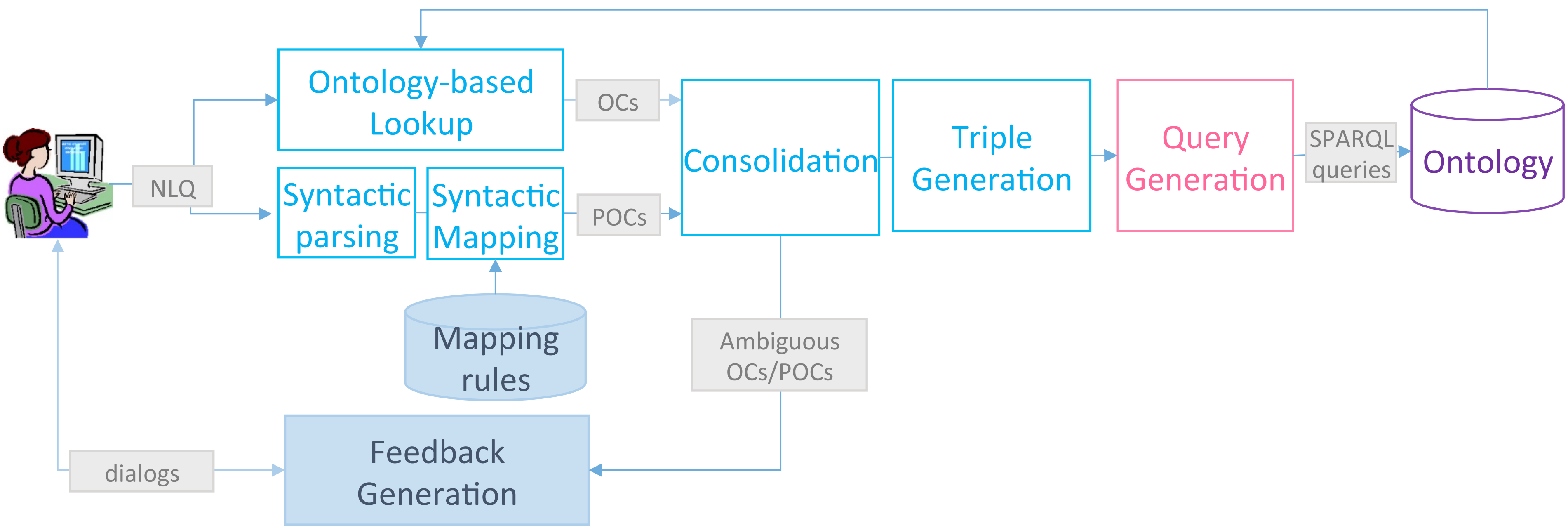
  - Ambiguity in join-path → leverage Schema-free XQuery to find out the optimal join-path

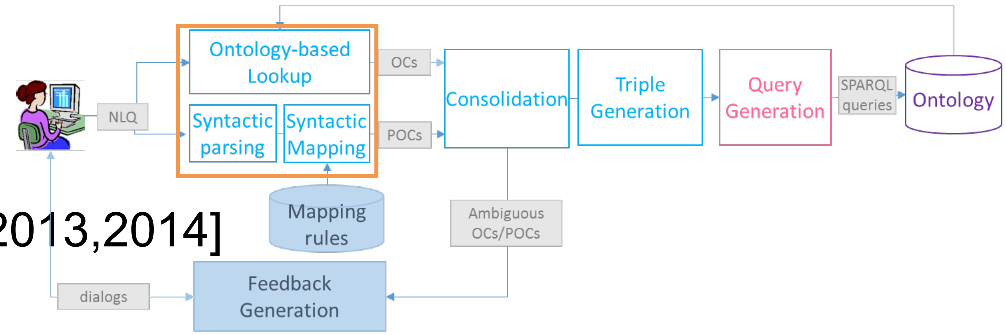
    e.g. There could be multiple ways for a *river* to be related to a *state*

- Error handling
  - Do not handle parser error explicitly
  - Interactive UI to encourage NLQ input understandable by the system

FREyA [Damljanovic et al., 2013,2014]

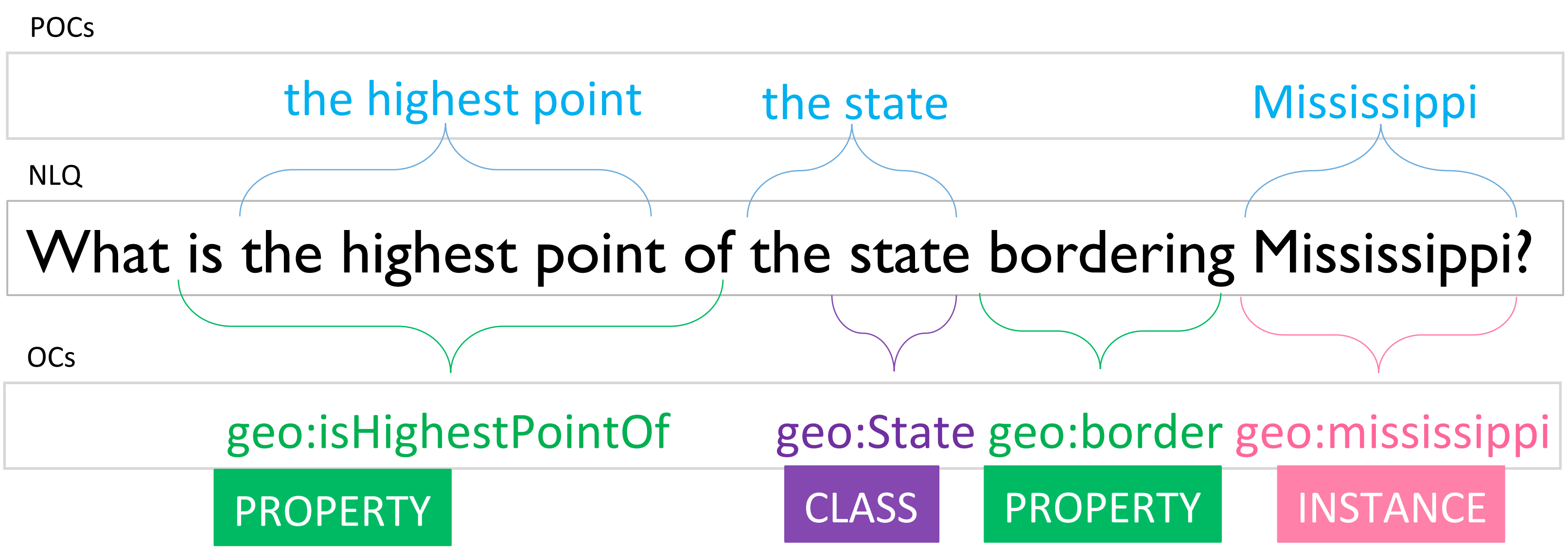
- Support ad-hoc NLQs, including ill-formed queries
  - Direct ontology look up + parse tree mapping → Certain level of robustness

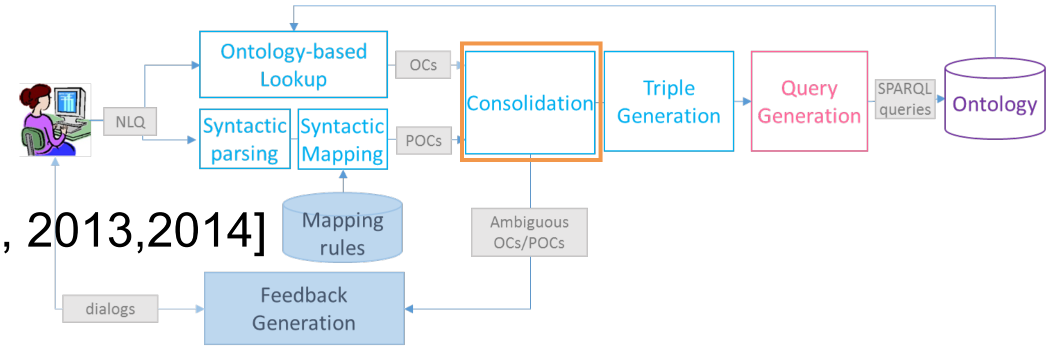




FREyA [Damljanovic et al., 2013,2014]

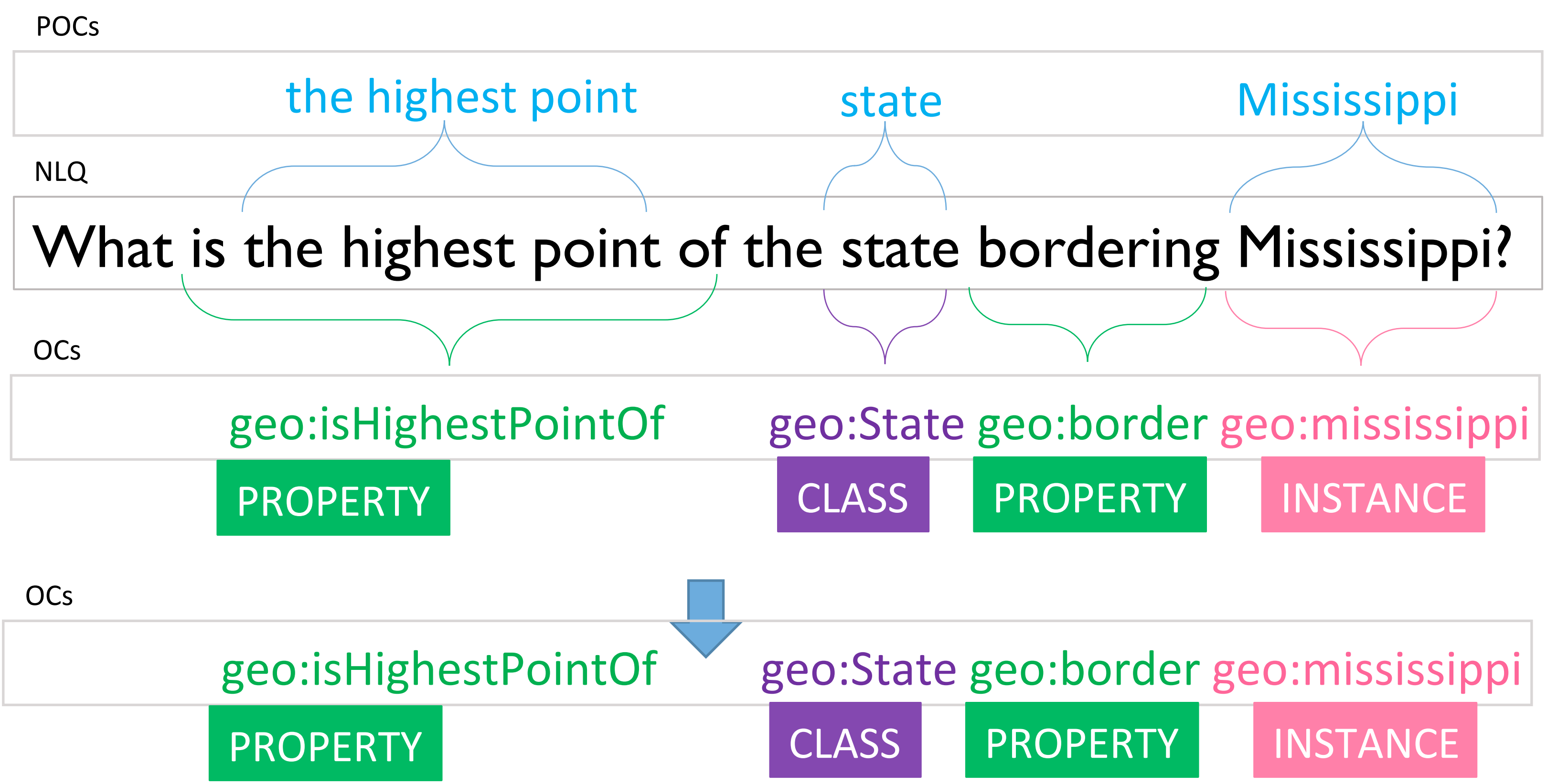
- Parse tree mapping based on pre-defined heuristic rules
  → Finds POCs (Potential Ontology Concept)
- Direct ontology look up
  → Finds OCs (Ontology Concept)





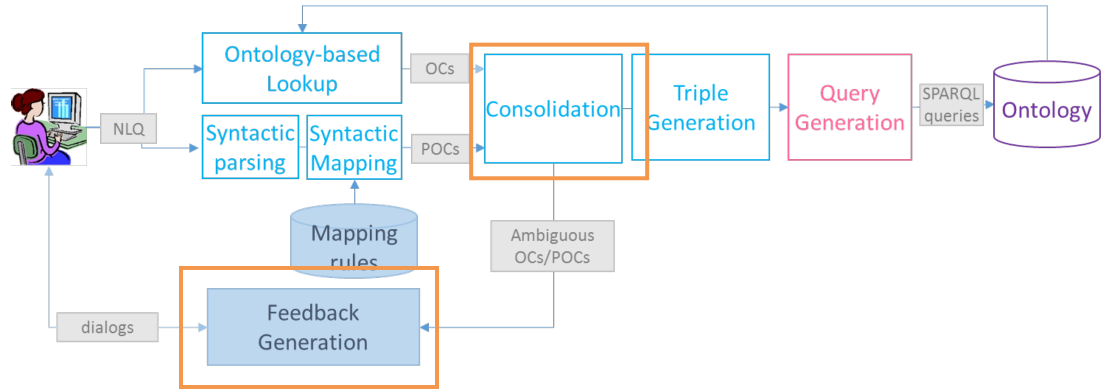
FREyA [Damljanovic et al., 2013,2014]

- Consolidate POCs and OCs
  - If span(POC) ⊆ span(OC) → Merge POC and OC



FREyA

[Damljanovic et al., 2013,2014]



- Consolidate POCs and OCs
  - If span(POC) ⊆ span(OC) → Merge POC and OC
  - Otherwise, provide suggestions and ask for user feedback

POCs

population California

NLQ

Return the population of California

OCs

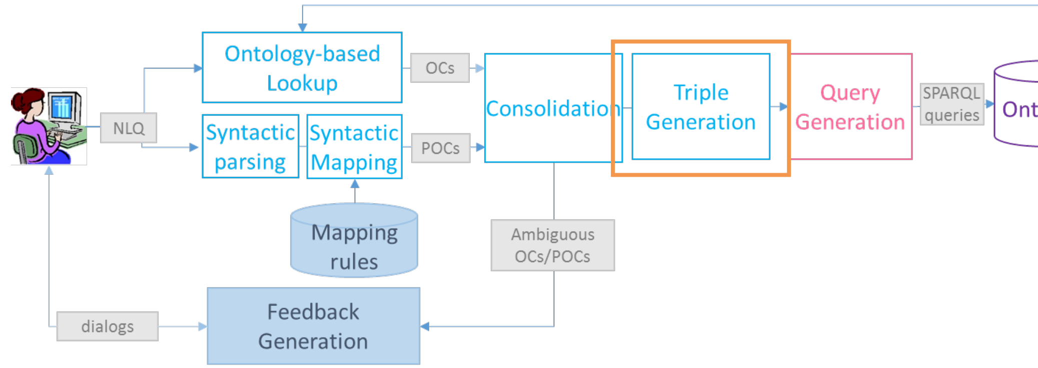
geo:california

INSTANCE

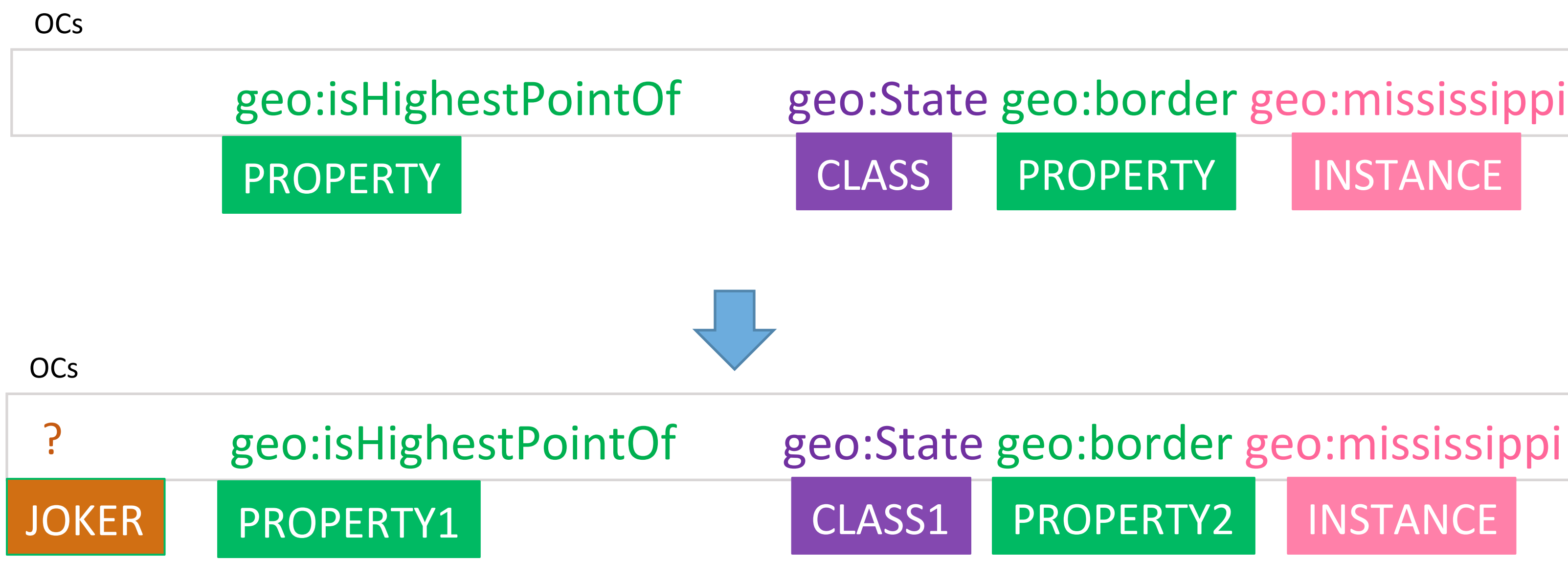
Suggestions ranked based on string similarity (Monge Elkan + Soundex)

1. state population 2. state population density 3. has low point, …

FREyA
[Damljanovic et al., 2013,2014]

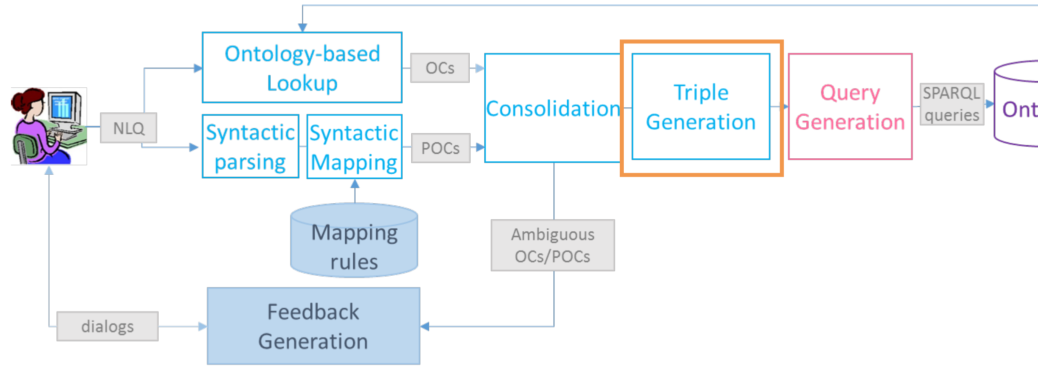


- Triple Generation: (1) Insert *joker* class



FREyA

[Damljanovic et al., 2013,2014]



- Triple Generation: (2) Generate triples

OCs

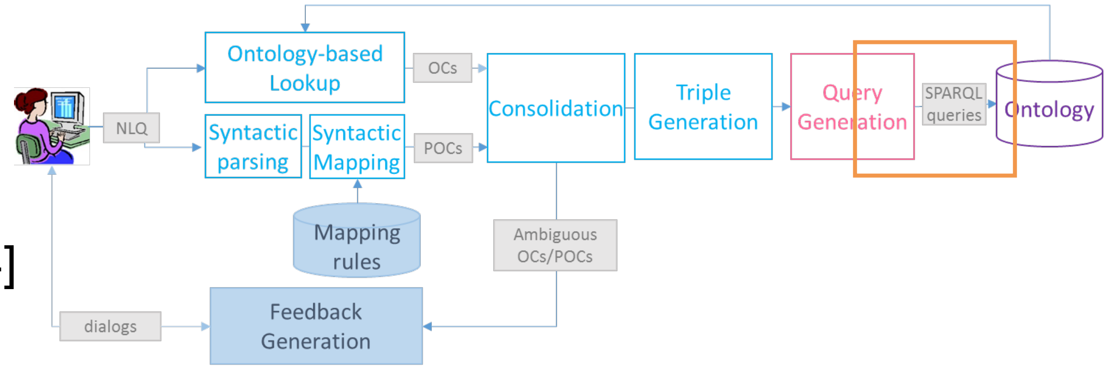
?	geo:isHighestPointOf	geo:State	geo:border	geo:mississippi
JOKER	PROPERTY1	CLASS1	PROPERTY2	INSTANCE

Triples

```
? - geo:isHighestPointOf - geo:State;
geo:State - geo:borders - geo:mississippi (geo:State);
```

FREyA

[Damljanovic et al., 2013,2014]



- Generate SPARQL query

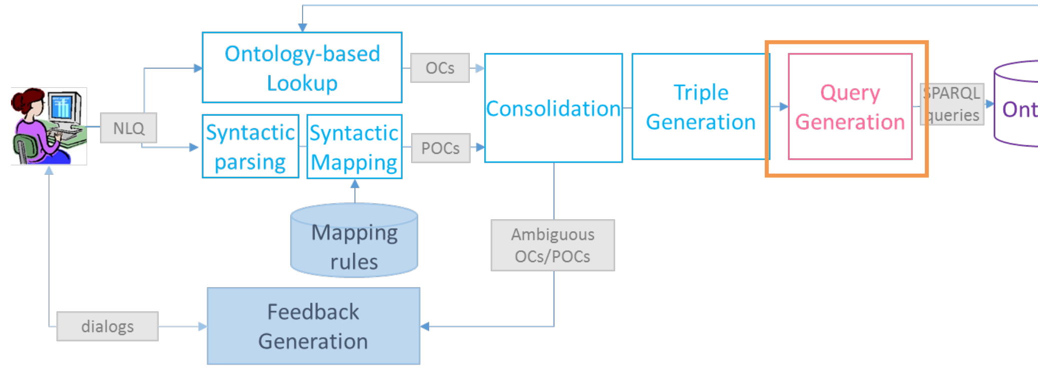
Triples

```
? - geo:isHighestPointOf - geo:State;
geo:State - geo:borders - geo:mississippi (geo:State);
```

```
prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
prefix geo: <http://www.mooney.net/geo#>
select ?firstJoker ?p0 ?c1 ?p2 ?i3
where { { ?firstJoker ?p0 ?c1 .
filter (?p0=geo:isHighestPointOf) . }
?c1 rdf:type geo:State .
?c1 ?p2 ?i3 .
filter (?p2=geo:borders) .
?i3 rdf:type geo:State .
filter (?i3=geo:mississippi) . }
```

FREyA

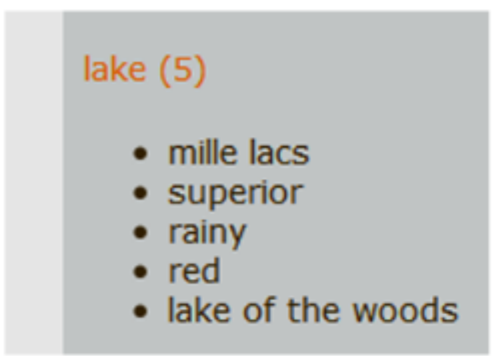
[Damljanovic et al., 2013,2014]



- Determine return type
  - Result of a SPARQL query is a graph
  - Identify answer type to decide the result display

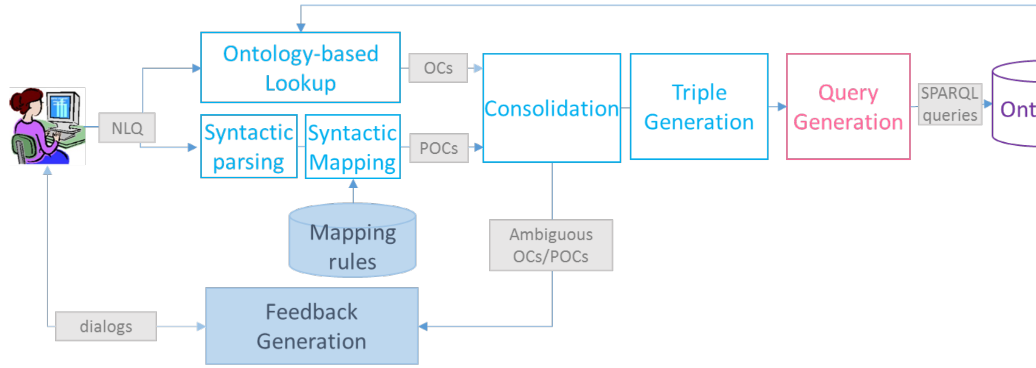
NLQ

Show lakes in Minnesota.



FREyA

[Damljanovic et al., 2013,2014]

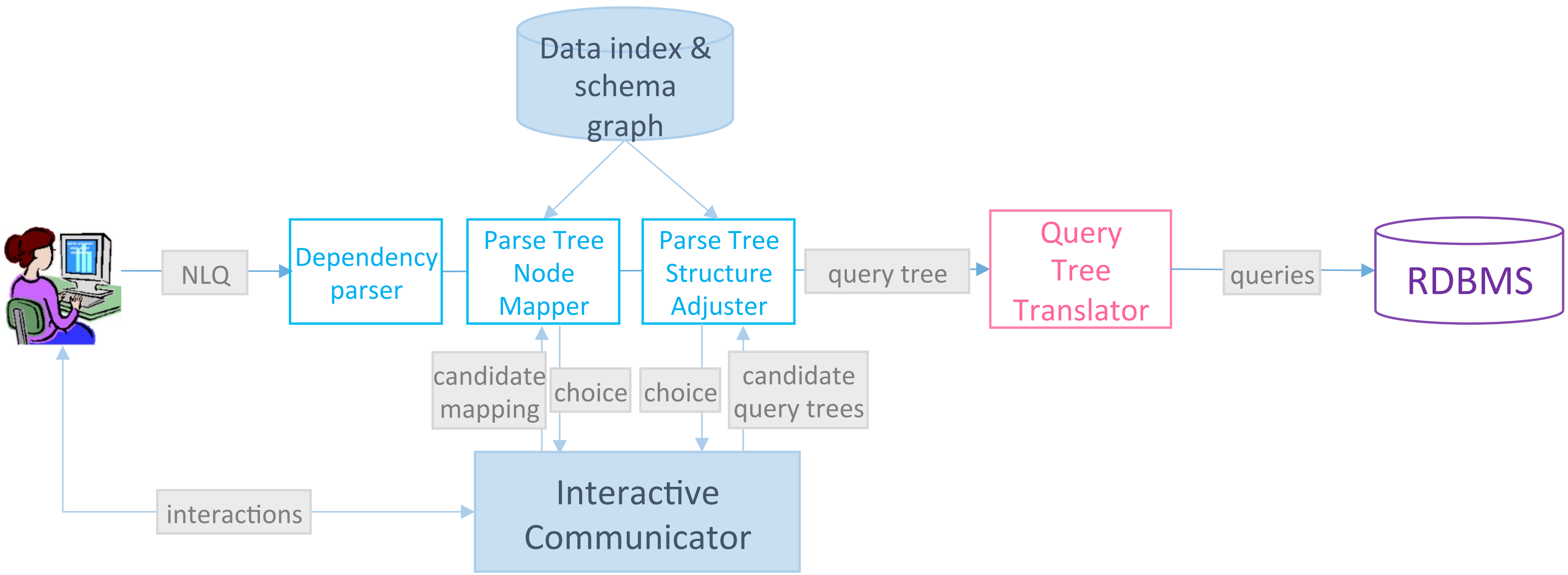


- Handle ambiguities via user interactions
  - Provide suggestions
  - Leverage re-enforcement learning to improve ranking of suggestions

- No parser error handling

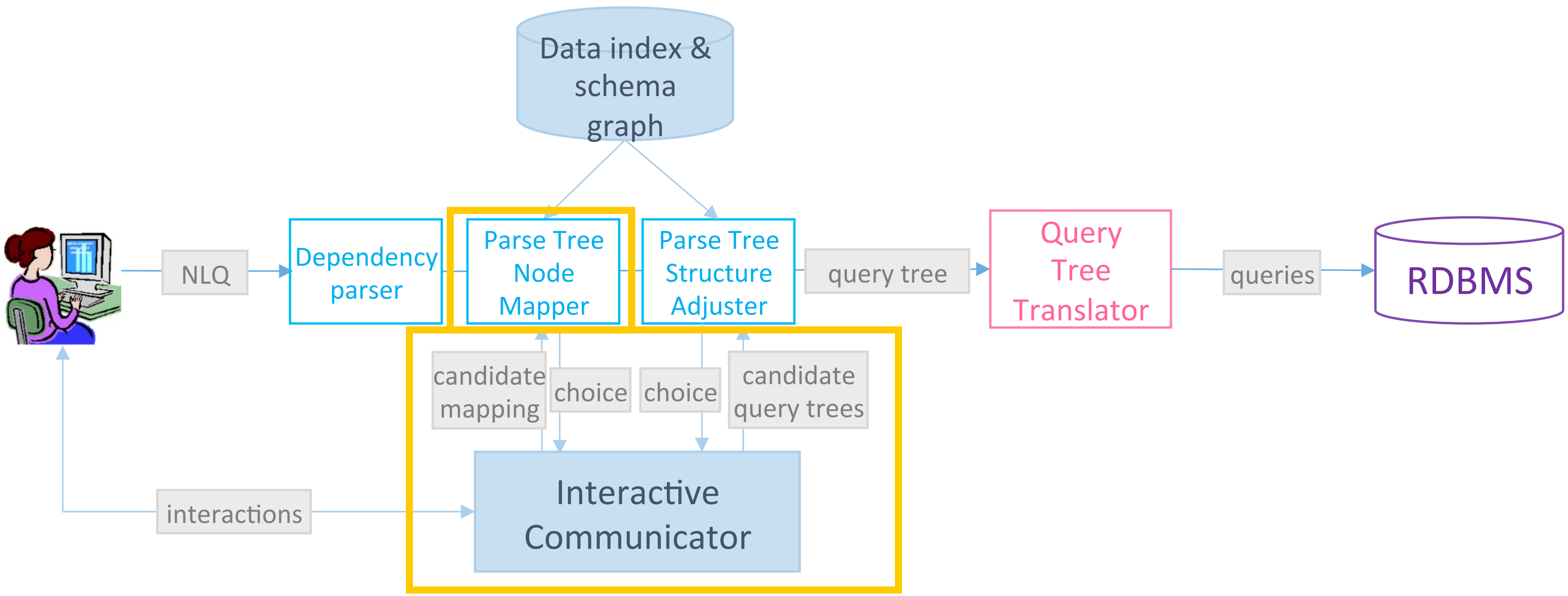
NaLIR [Li and Jagadish, 2014]

- Controlled NLQ based on predefined grammar
- No query history



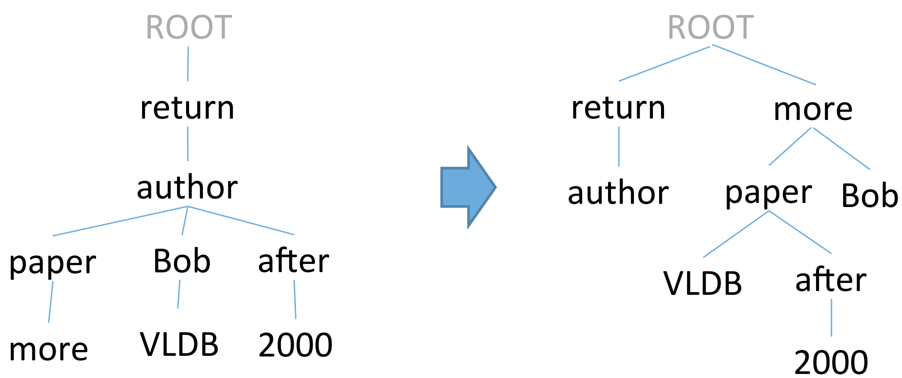
NaLIR [Li and Jagadish, 2014]

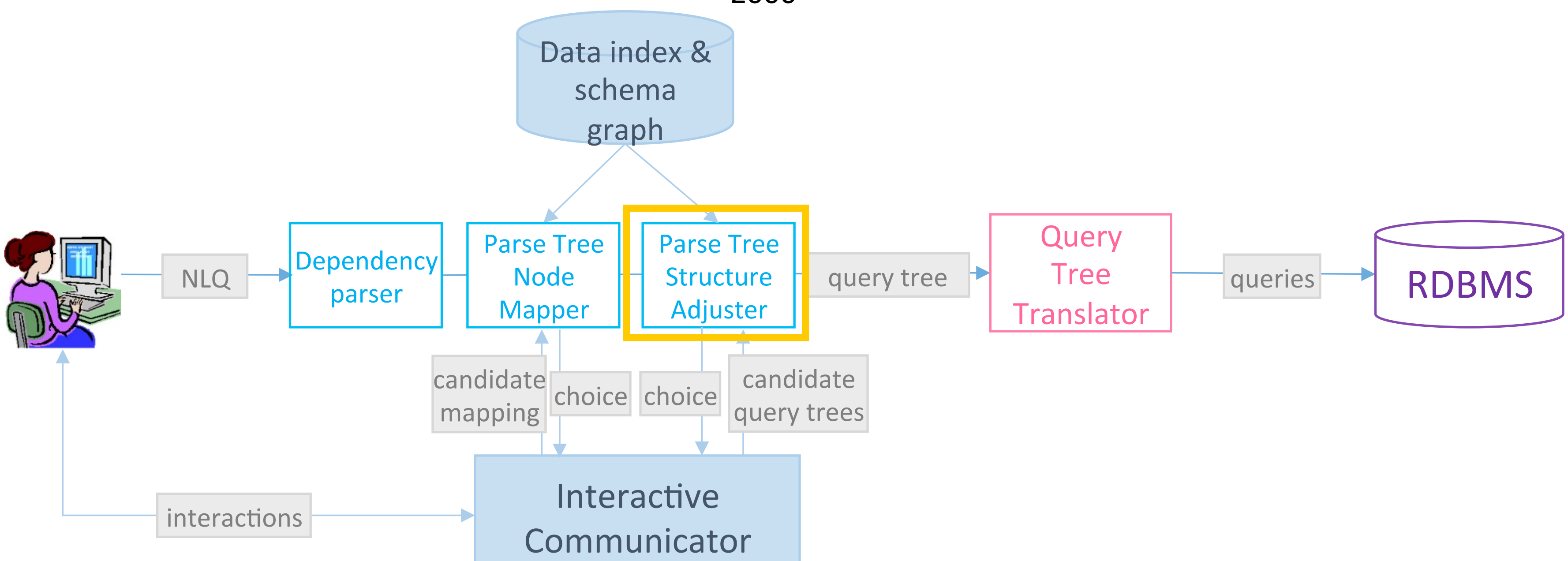
- Mapping parse tree node to data schema and value based on WUP similarity [Wu and Palmer, 1994]
- Explicitly request user input on ambiguous mappings and interpretations



NaLIR [Li and Jagadish, 2014]

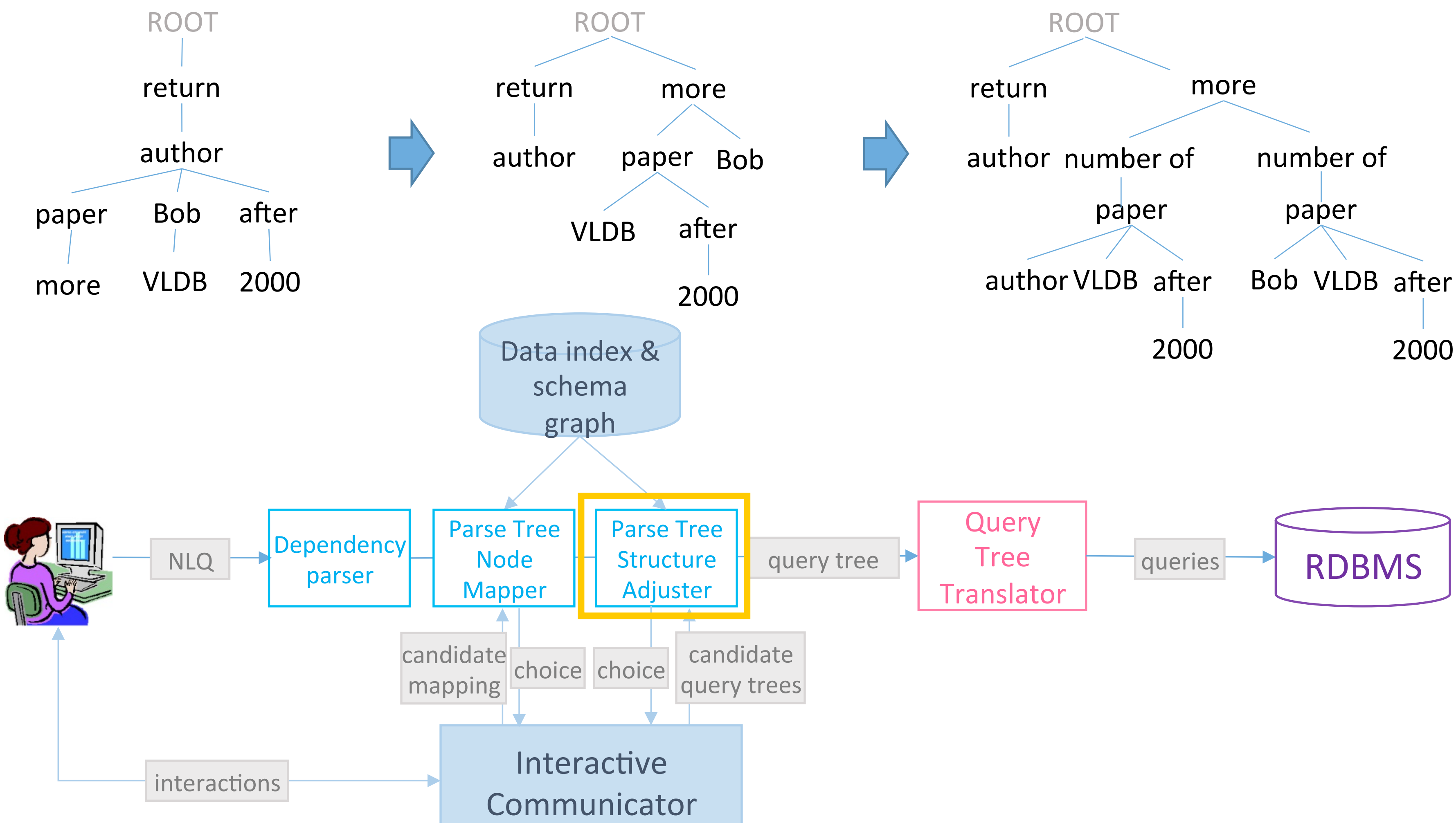
- Automatically adjust parse tree structure into a valid parse tree





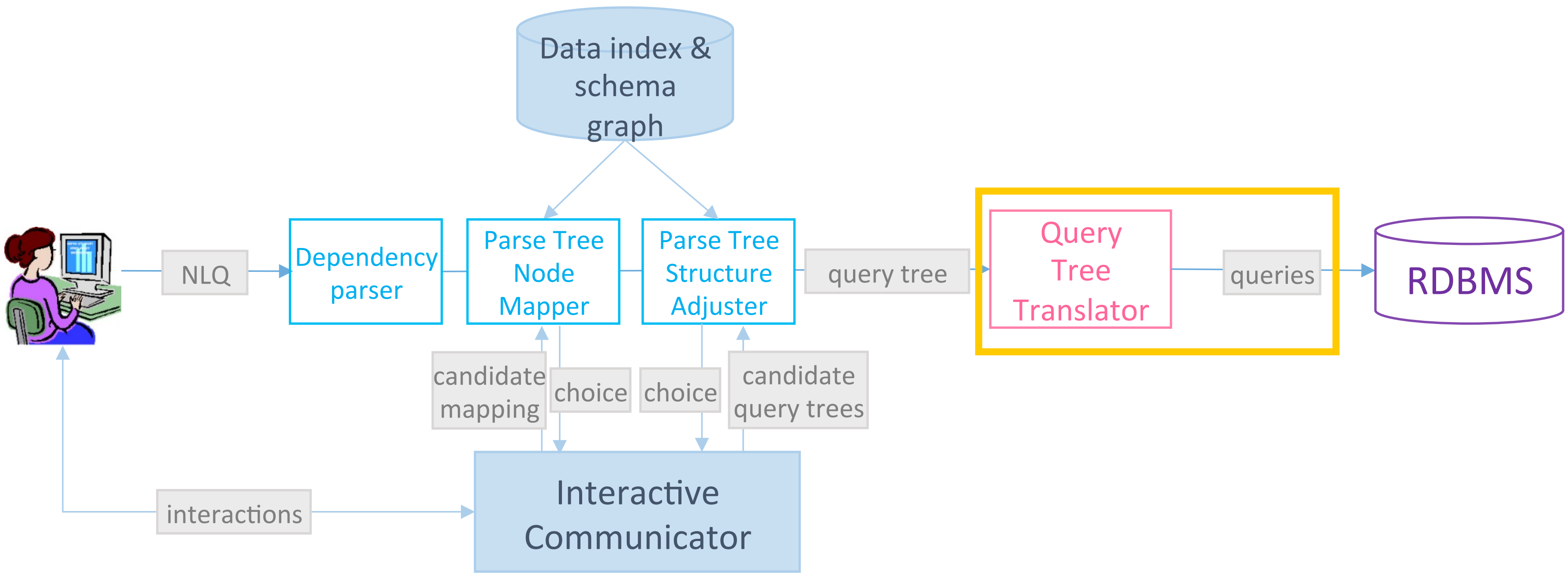
NaLIR [Li and Jagadish, 2014]

- Automatically adjust parse tree structure into a valid parse tree
- Further rewrite parse tree into one semantically reasonable



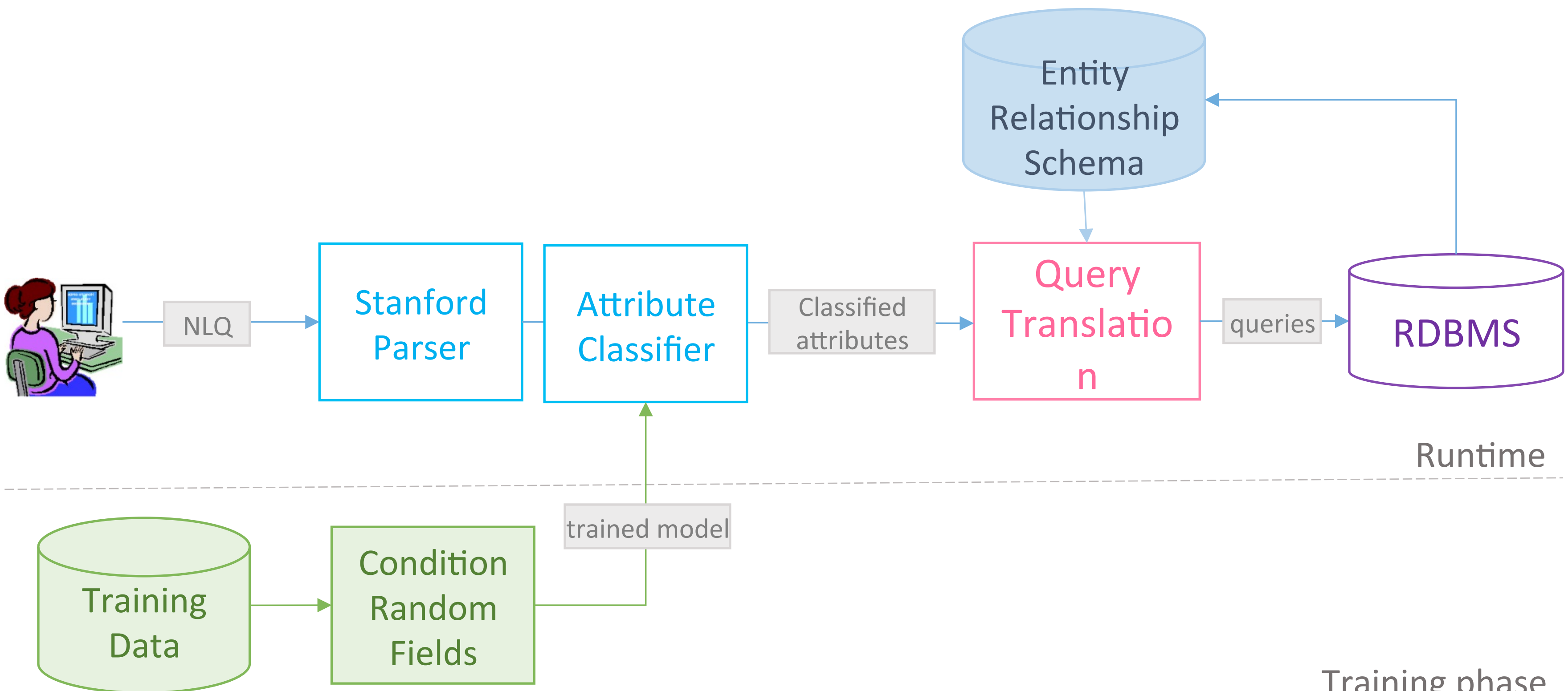
NaLIR [Li and Jagadish, 2014]

- 1-1 translation from query tree to SQL



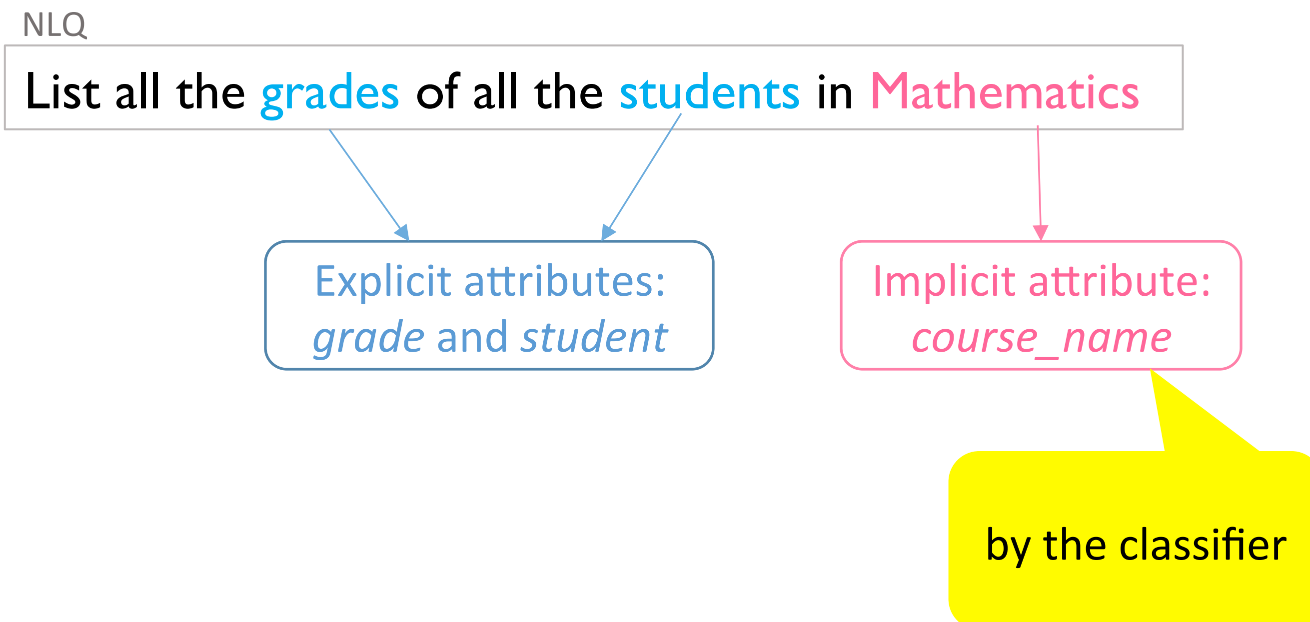
Learning NLQ → SQL [Palakurthi et al., 2015]

- Ad-hoc NLQ queries with explicit attribute mentions
  - Implicit restriction imposed by the capability of the system itself

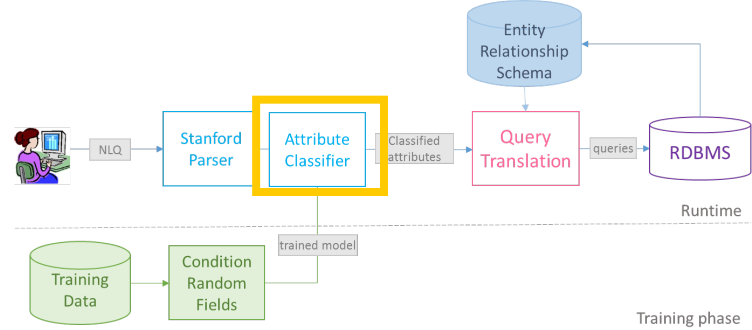


Learning NLQ → SQL [Palakurthi et al., 2015]

- Explicit attributes: attributes mentioned explicitly in the NLQ



Learning NLQ → SQL
[Palakurthi et al., 2015]



- Learn to map explicit attributes in the NLQ to SQL clauses

Training data

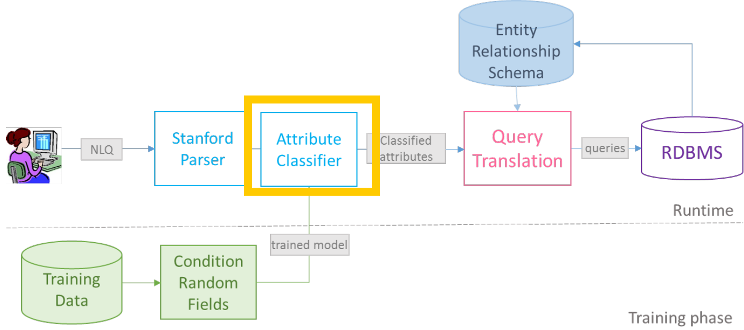
Token	Attribute	Tag
What	0	O
are	0	O
the	0	O
courses	1	GROUP BY
with	0	O
less	0	O
than	0	O
25	0	O
students	1	HAVING
?	0	O

Features

Type of Feature	Example Feature
Token-based	isSymbol
Grammatical	POS tags and grammatical relations
Contextual	Tokens preceding or following the current token
Other	• isAttribute • Presence of other attributes • Trigger words (e.g. "*each*")

Learning NLQ → SQL

[Palakurthi et al., 2015]



- Learn to map explicit attributes in the NLQ to SQL clauses

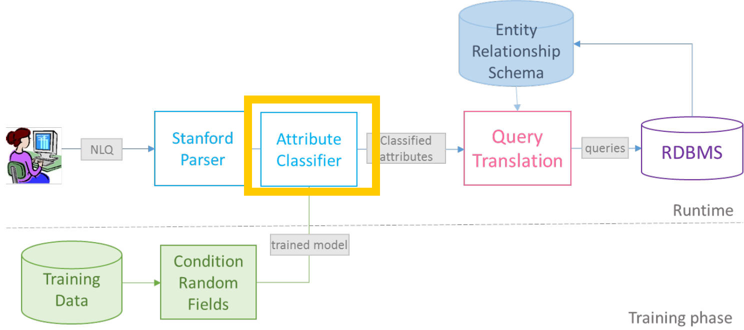
GROUP BY — FROM — HAVING

NLQ

Who are the professors teaching more than 2 courses?

Learning NLQ → SQL

[Palakurthi et al., 2015]



- Construct full SQL queries
  - Attribute→ Clause Mapping
  - Identify joins based on ER diagram
  - Add missing implicit attributes via Concept Identification [Srirampur et al., 2014]

GROUP BY FROM HAVING

NLQ

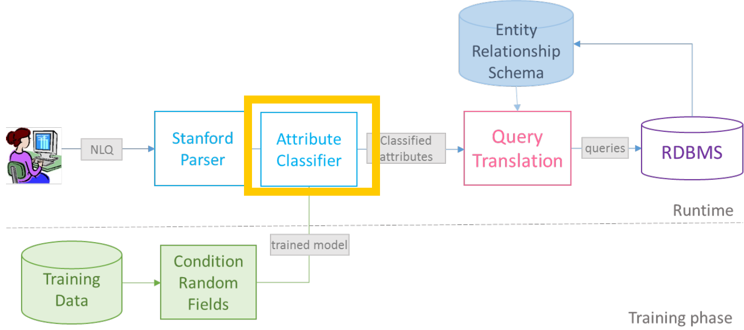
Who are the professors teaching more than 2 courses?

SQL

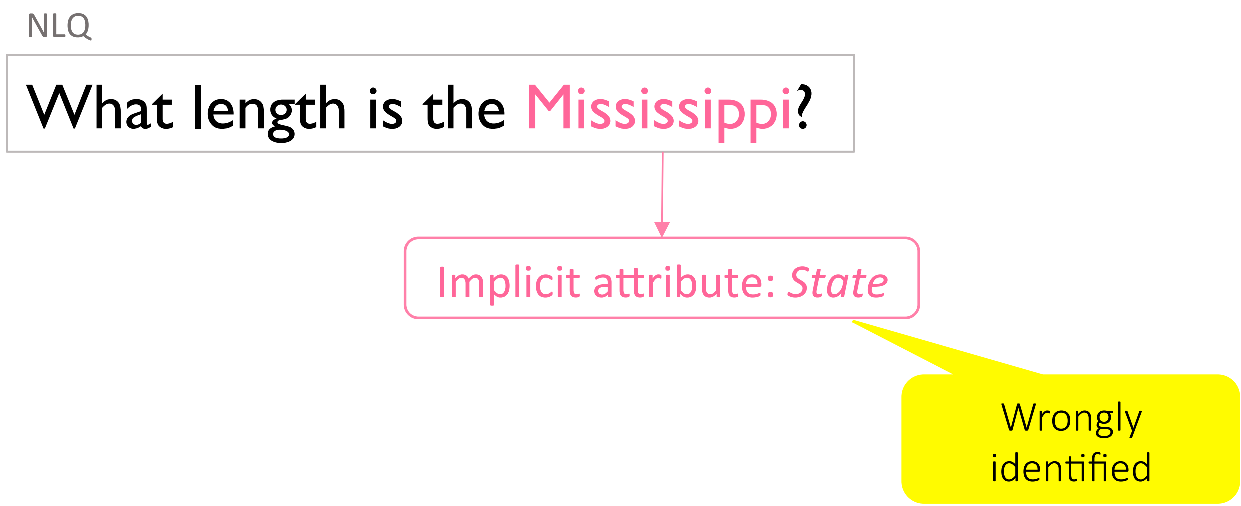
```
SELECT professor_name
FROM   COURSES,TEACH,PROFESSOR
WHERE  course_id=course_teach_id
  AND  prof_teach_id =prof_id
GROUP BY professor_name
HAVING COUNT(course_name) > 2
```

Identified based ER schema

Learning NLQ → SQL
[Palakurthi et al., 2015]

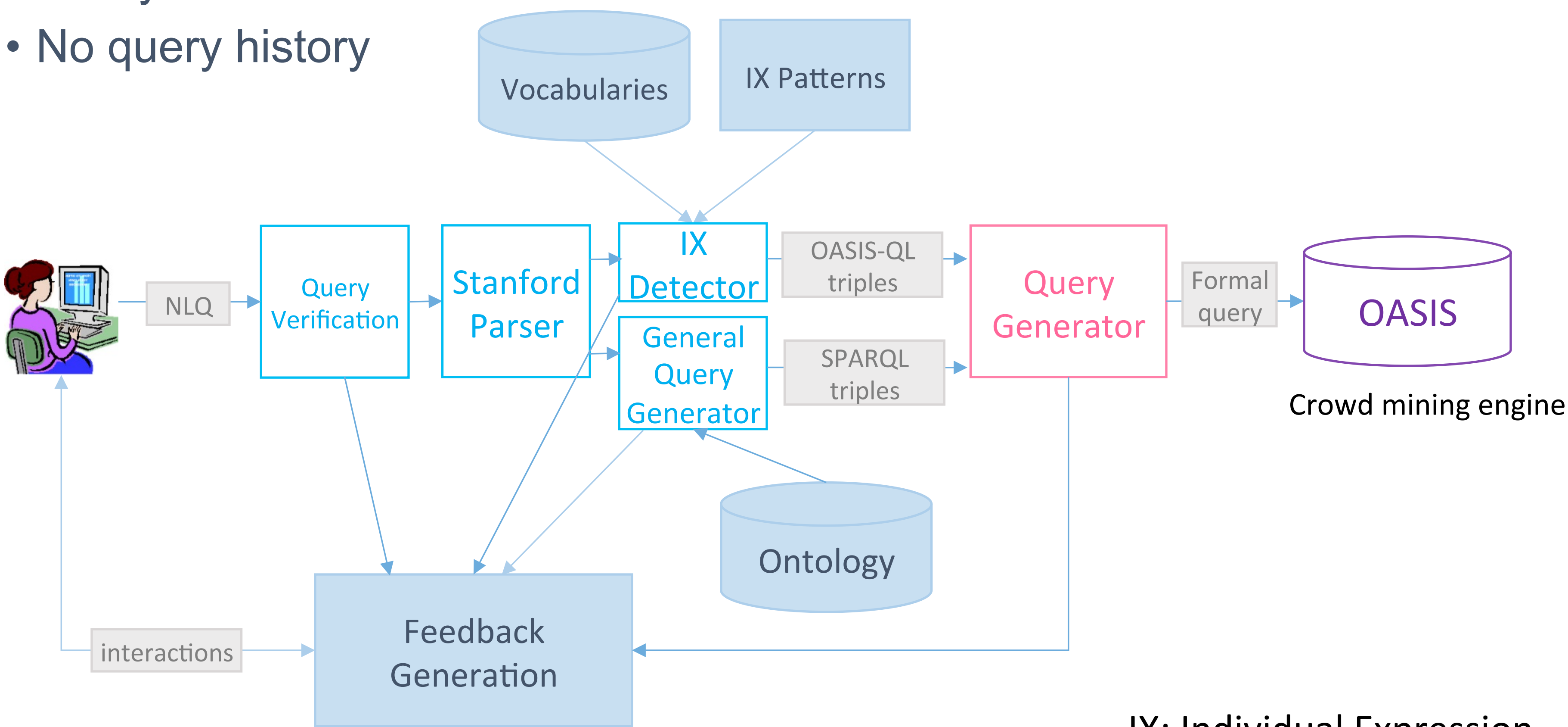


- No parsing error handling
- No explicit ambiguity handling



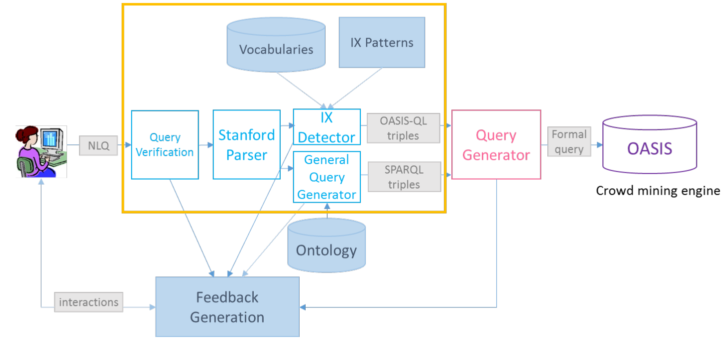
NL_2CM [Amsterdamer et al., 2015]

- Controlled NLQ based on predefined types (e.g. no "why" questions)
- Query verification with feedback
- No query history



IX: Individual Expression

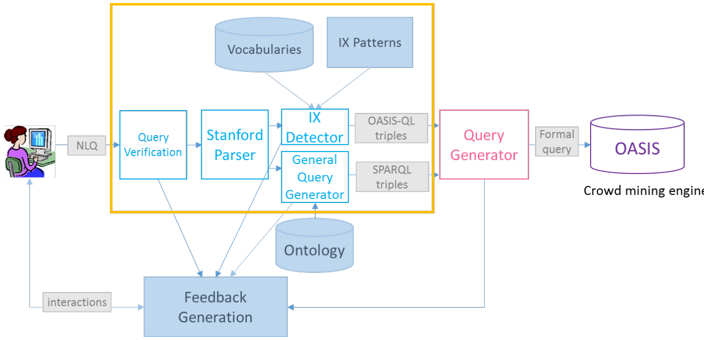
NL$_2$CM [Amsterdamer et al., 2015]



- Map parse tree with **I**ndividual E**x**pression (IX) patterns and vocabularies
  - **Lexical individuality:** Individual terms convey certain meaning
  - **Participant individuality:** Participants or agents in the text that that are relative to the person addressed by the request
  - **Synctatic individuality:** Certain syntactic constructs in a sentence.

What are the most interesting places near Forest Hotel, Buffalo that we should visit?

NL$_2$CM [Amsterdamer et al., 2015]



- Map parse tree with **I**ndividual E**x**pression (IX) patterns and vocabularies
  - **Lexical individuality:** Individual terms convey certain meaning
  - **Participant individuality:** Participants or agents in the text that that are relative to the person addressed by the request
  - **Synctatic individuality:** Certain syntactic constructs in a sentence.

What are the most interesting places near Forest Hotel, Buffalo that we should visit?

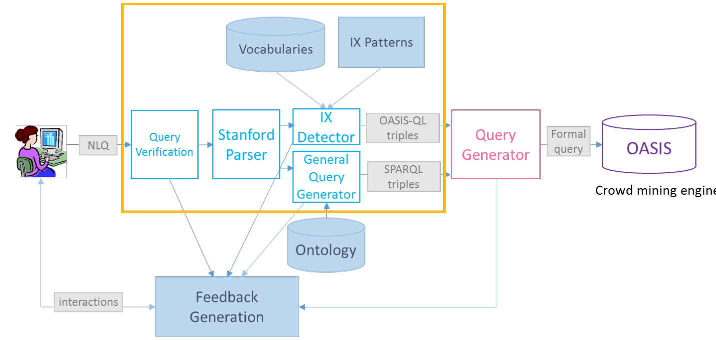
Opinion Lexicon

$x interesting

```
$x subject $y
filter(POS($x) = "verb" && $y in V_participant)
```

[] visit $x

NL$_2$CM [Amsterdamer et al., 2015]



- Map parse tree with **I**ndividual E**x**pression (IX) patterns and vocabularies
- Processing the general parts of the query with FREyA system
- Interact with user to resolve ambiguities

What are the most interesting places near Forest Hotel, Buffalo that we should visit?

Opinion Lexicon

User interaction

```
$x subject $y
filter(POS($x) = "verb" && $y in V_participant)
```

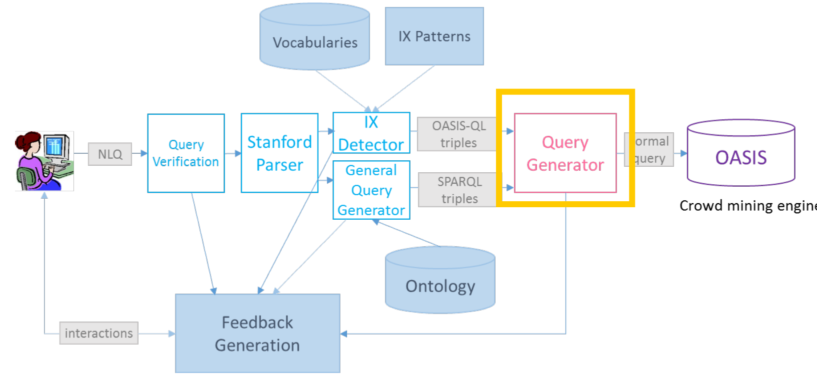
$x interesting

$x near Forest Hotel,_Buffalo,_NY

[] visit $x

$x instanceOf Place

NL$_2$CM [Amsterdamer et al., 2015]



- No parsing error handling
- Return error for partially interpretable queries
- SPARQL + OASIS-QL triples → a complete OASIS-QL query

$x near Forest Hotel,_Buffalo,_NY

$x instanceOf Place

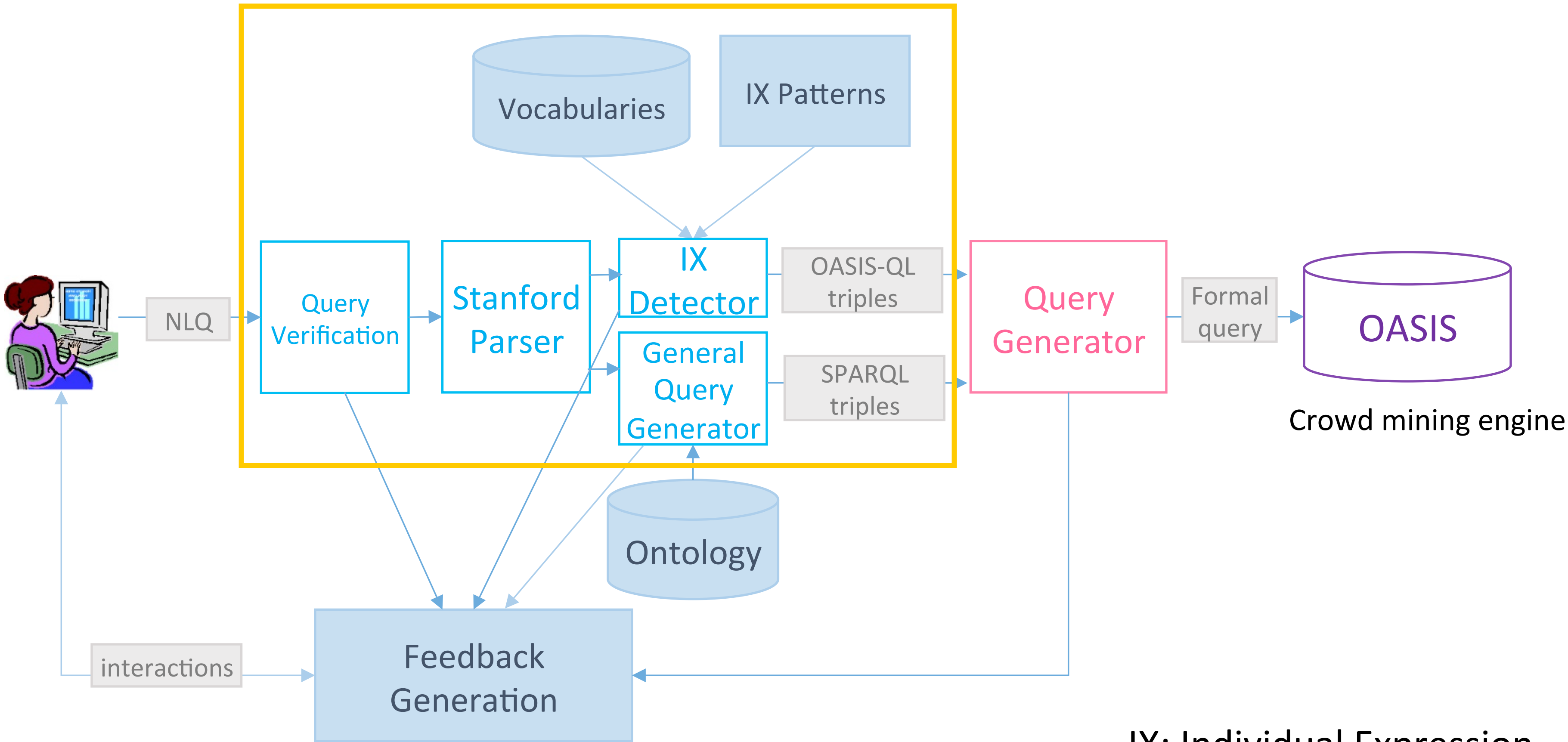
$x interesting

[] visit $x

```
SELECT VARIABLES
WHERE
    {$x instanceOf Place.
     $x near      Forest_Hotel,_Buffalo,_NY}
SATISFYING
    {$x hasLabel   "interesting"}
     ORDER BY DESC(SUPPORT)
     LIMIT 5
    AND
     { [ ] visit $x}
       WITH SUPPORT THRESHOLD = 0.1
```

NL_2CM [Amsterdamer et al., 2015]

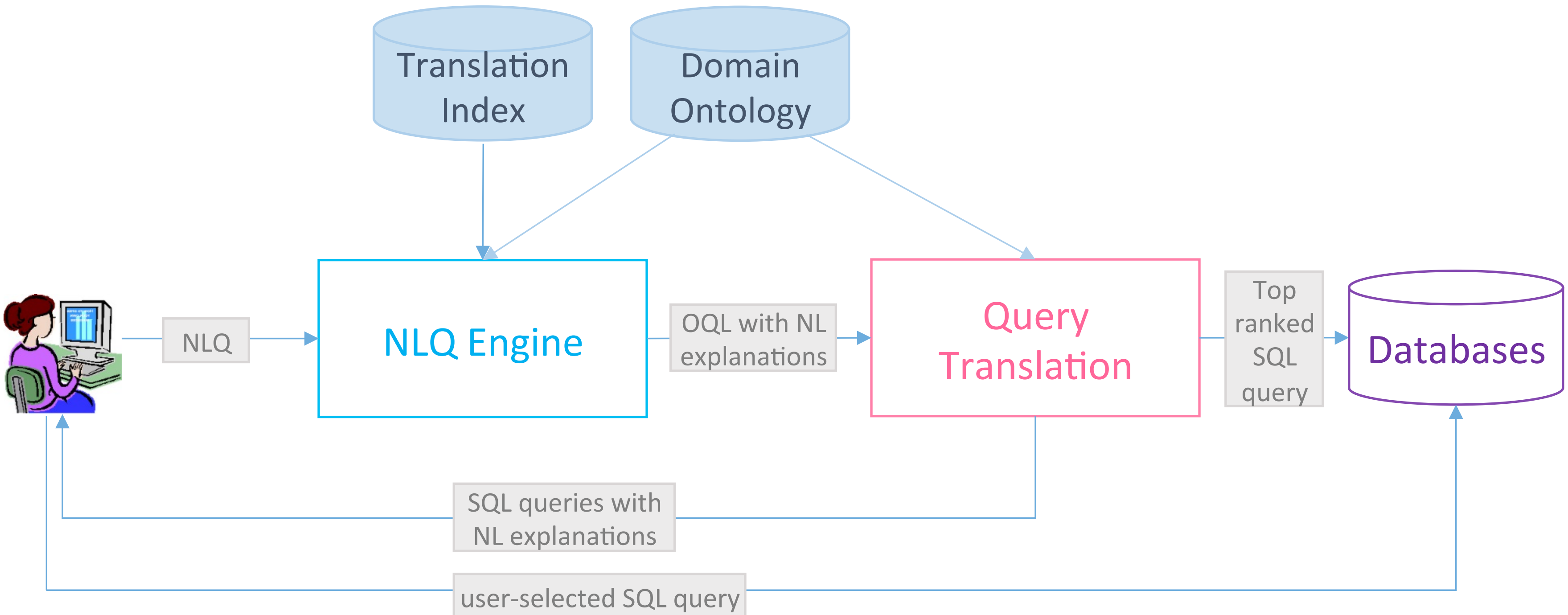
- Handling ambiguity via user input



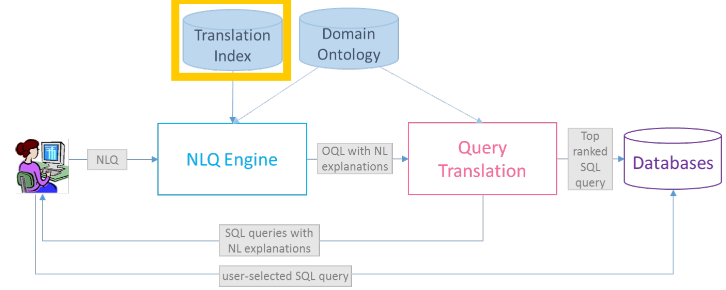
IX: Individual Expression

ATHANA [Saha et al., 2016]

- Permit ad-hoc queries
  - No explicit constraints on NLQ
  - Implicit limit on expressivity of NLQs by query expressivity limitation (e.g. nested query with more than 1 level)
- No query history



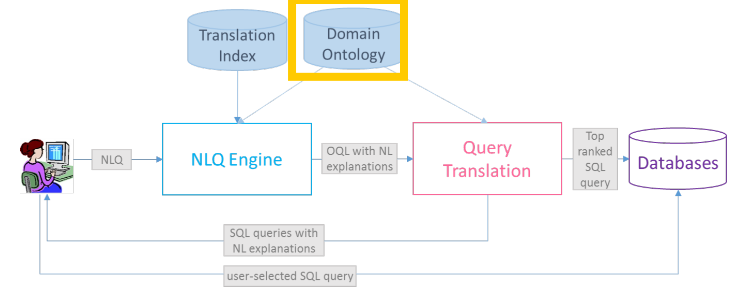
ATHANA [Saha et al., 2016]

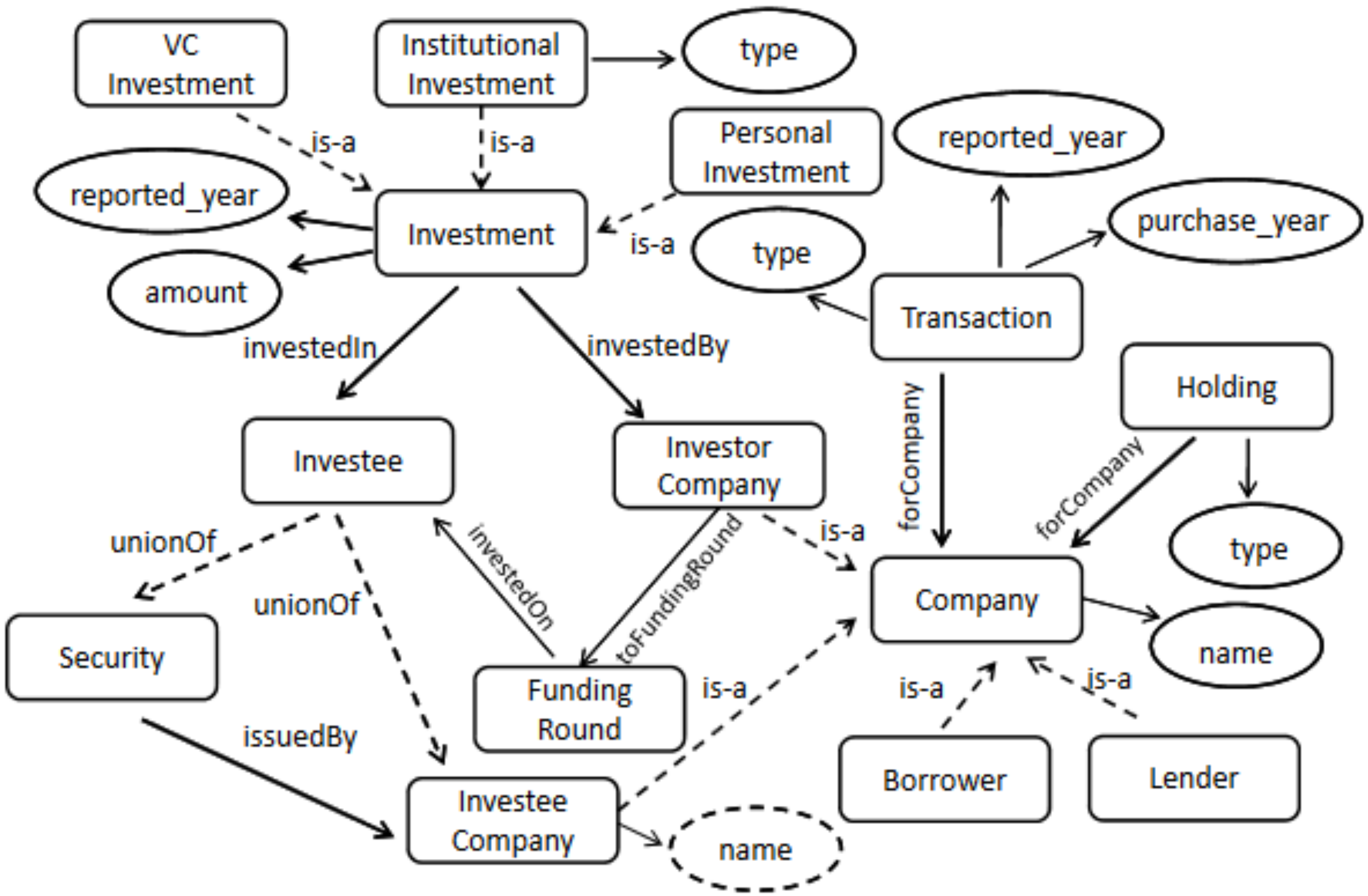


- Annotate NLQ into evidences → No explicit parsing
- Handle ambiguity based on translation index and domain ontology

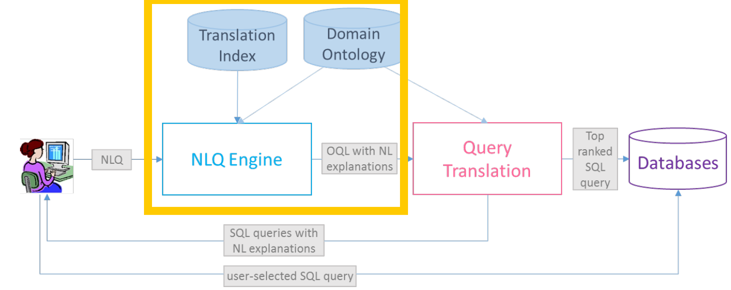
Translation Index

ATHANA [Saha et al., 2016]





ATHANA [Saha et al., 2016]



Show me restricted stock investments in Alibaba since 2012 by year

restricted stock → indexed value; investments → metadata; Alibaba → indexed value; since 2012 → time range; year → metadata

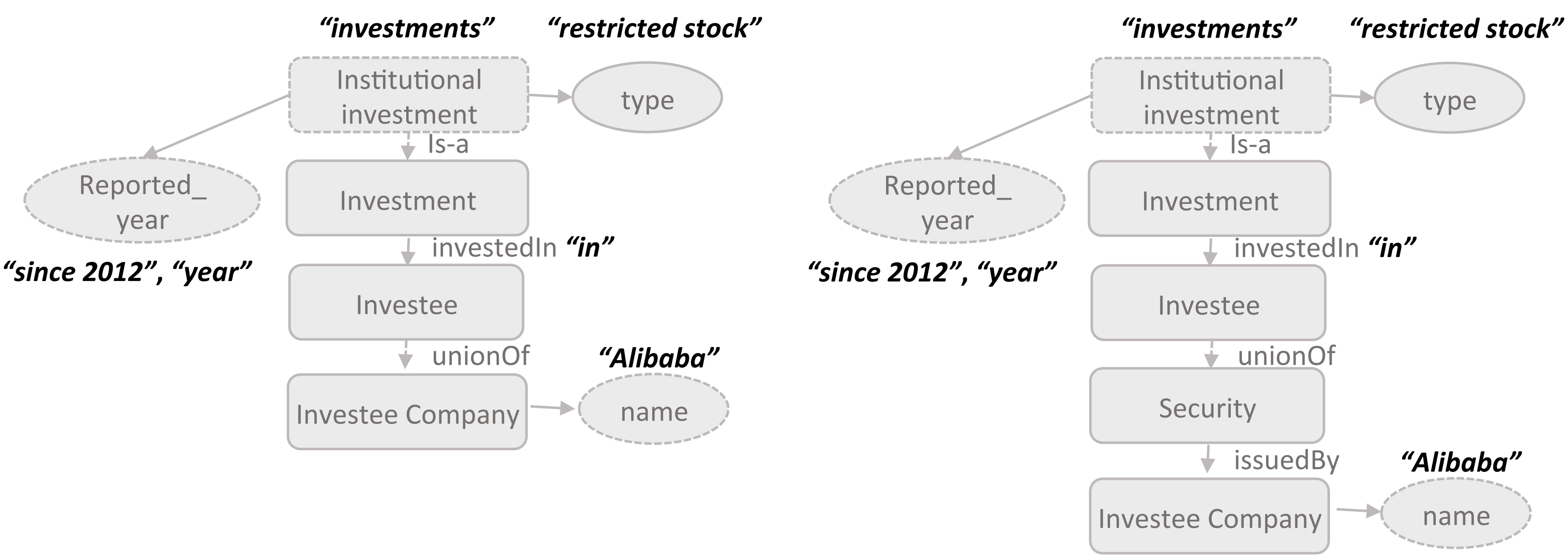
Evidence

Holding.type
Transaction.type
InstitutionalInvestment.type
…

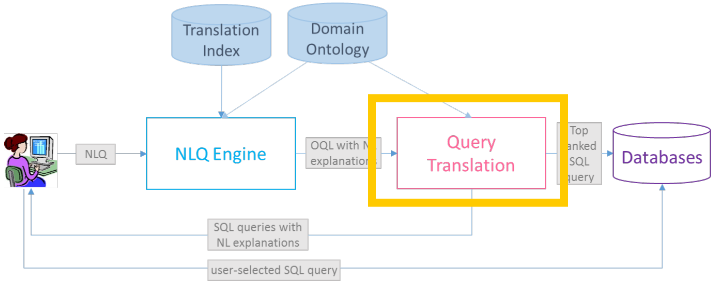
PersonalInvestment
InstitutionalInvestment
VCInvestment
…

Company.name:Alibaba Inc.
Company.name:Alibaba Holding Inc.
…

Transaction.reported_year
Transaction.purchase_year
InstitutionalInvestment.reported_year
…



Interpretation trees

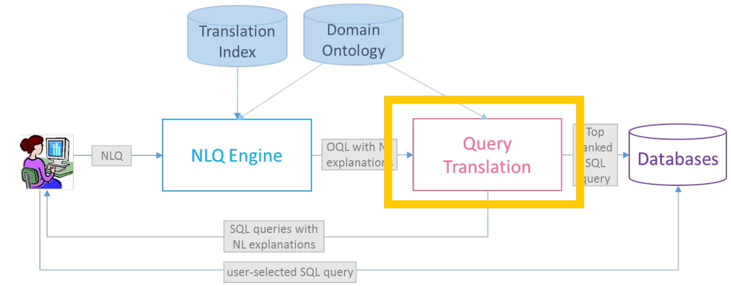


ATHANA [Saha et al., 2016]

- Ontology Query Language
  - Intermediate language over domain ontologies
  - Separate query semantics from underlying data stores
  - Support common OLAP-style queries

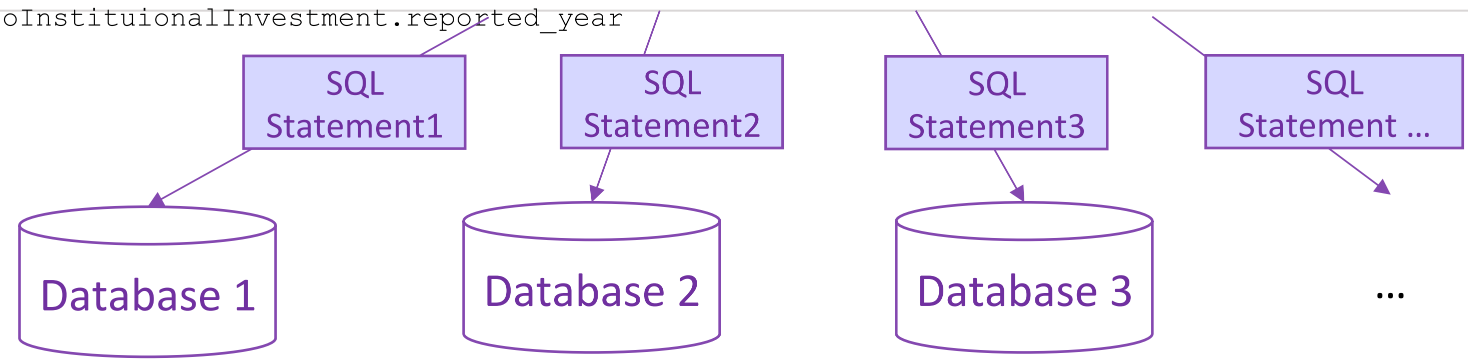
```
UnionQuery:    Query (UNION Query)*
Query:         select from where?  groupBy?  orderBy?  having?
select:        (aggrType?(PropertyRef))+
from:          (Concept ConceptAlias)+
where:         binExpr1* binExpr2* inExpr?
groupBy:       (PropertyRef)+
orderBy:       (aggrType?(PropertyRef))+
having:        aggrType(PropertyRef) binOp value
value:         Literal+ | Query
aggrType:      SUM| COUNT| AVG | MIN | MAX
binExpr1:      PropertyRef binOp [any] value
binExpr2:      ConceptAlias RelationRef+ = ConceptAlias
inExpr:        PropertyRef IN Query
binOp:         > | < | >= | <= | =
PropertyRef:   ConceptAlias.Property
RelationRef:   Relation ->
```

ATHANA [Saha et al., 2016]



- 1-1 translation from interpretation tree to OQL
- 1-1 translation from OQL to SQL per relational schema

```
SELECT   Sum(oInstituionalINvestment.amount),
         oInstitutionalInvestment.reported_year
FROM     InstitutionalInvestment OInstitutionalInvestment,
         InvesteeCompany oInvesteeCompany
WHERE    oInstitutionalInvestment.type = "restricted_stock",
         oInstitutionalInvestment.reported_year >= '2012'
         oInstitutionalInvestment.reported_year >= Inf,
         oInvesteeCompany.name = ('Alibaba Holdings Ltd.', 'Alibaba Inc.', 'Alibaba Capital
Partners'},
         oInstitionalInvestment→isa→InvestedIn→unionOf_Security→issuedBy=oInvesteeCompany
GROUP BY oInstituionalInvestment.reported_year
```



NLIDBs Summary

Systems	Scope of NLQ Support		Capability		State		Parsing Error Handling	
	Controlled	Ad-hoc*	Fixed	Self-improving	Stateless	Stateful	Auto-correction	Interactive-correction
PRECISE	✓		✓		✓		✓	
NLPQC	✓		✓		✓			
NaLIX	✓			✓		✓		✓
FREyA		✓		✓	✓			
NaLIR	✓		✓		✓			
NL_2CM	✓			✓	✓		✓	
ML2SQL		✓	✓		✓			
ATHANA		✓	✓		✓		N/A	N/A

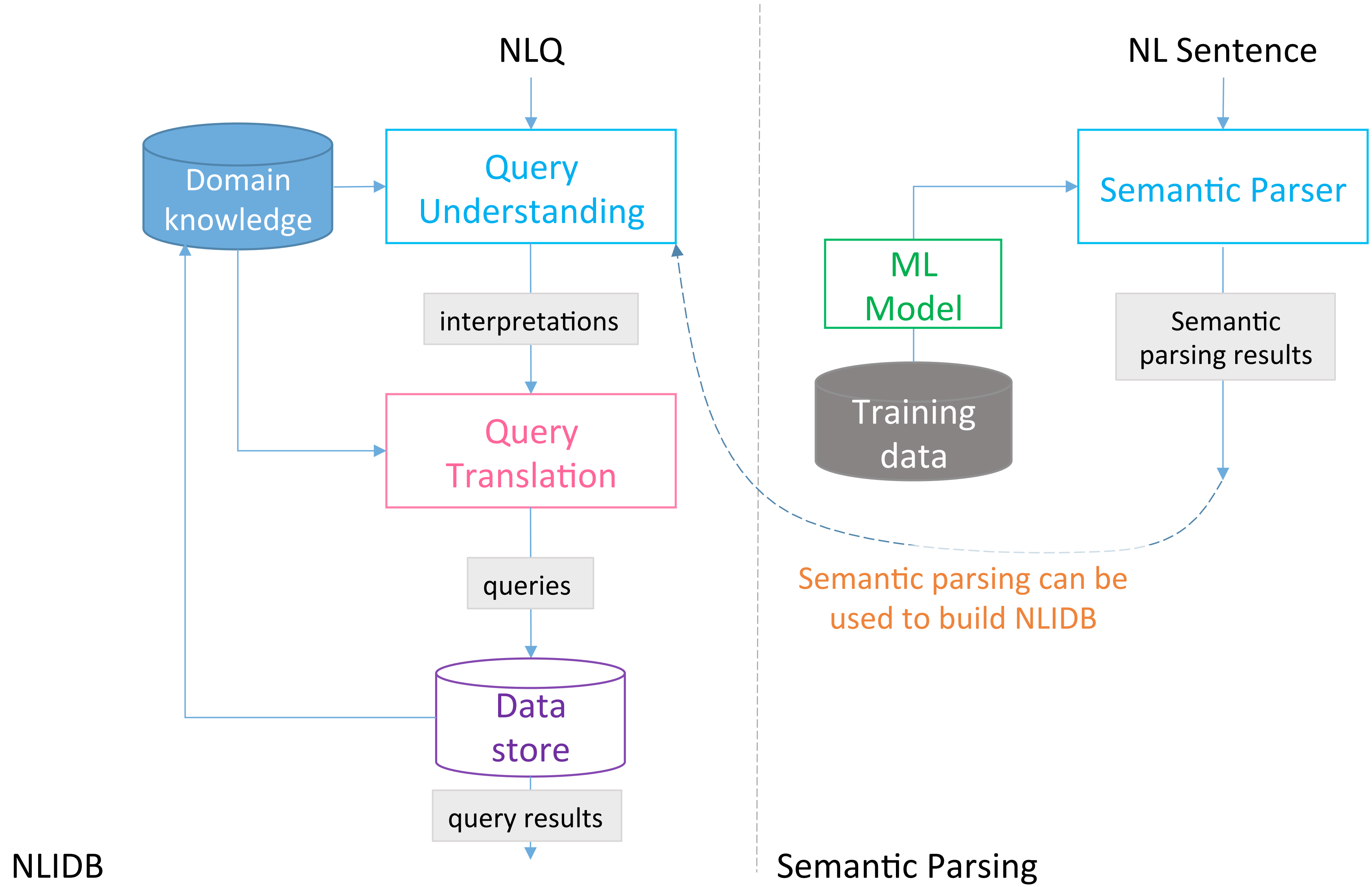
* Implicit limitation by system capability

NLIDBs Summary – Cont.

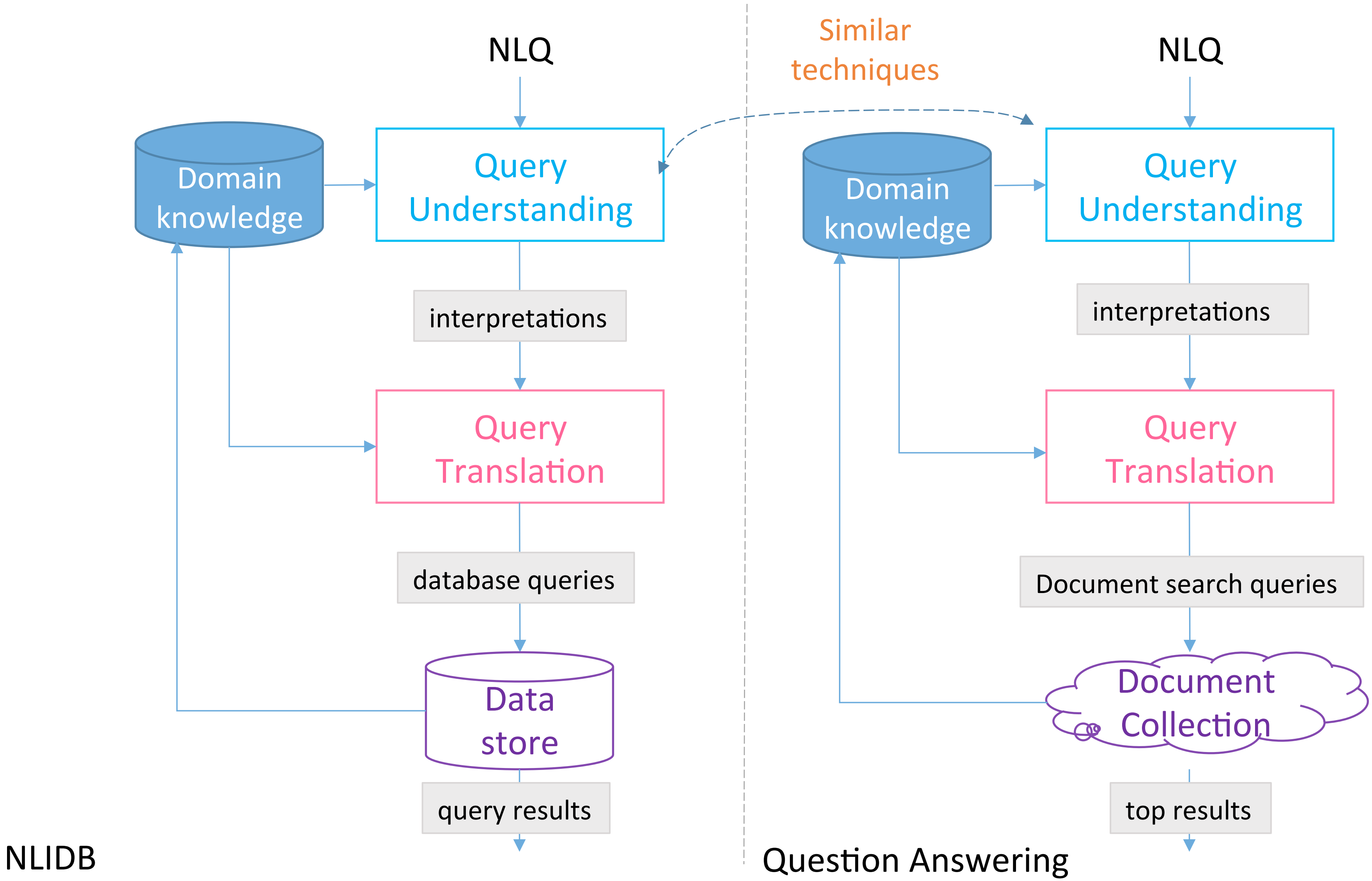
Systems	Ambiguity Handling		Query Construction		Target Language
	Automatic	Interactive	Rule-based	Machine-learning	
PRECISE	✓		✓		SQL
NLPQC			✓		SQL
NaLIX	✓	✓			(Schema-free) XQuery
FREyA		✓	✓		SPARQL
NaLIR		✓	✓		SQL
NL_2CM		✓	✓		OASIS-QL
ML2SQL				✓ *	SQL
ATHANA	✓		✓		OQL

* Partially

Relationship to Semantic Parsing



Relationship to Question Answering



Open Challenges and Opportunities

Querying Natural Language Data - Review

- Covered
  - Boolean queries
  - Grammar-based schema and searches
  - Text pattern queries
  - Tree pattern queries

- Developments beyond
  - Keyword searches as input
  - Documents as output

Querying Natural Language Data – Challenges & Opportunities

- Grammar-based schemas
  - Promising direction
- Challenges
  - Queries w/o knowing the schema
  - Many table schemes!
  - Overlap and equivalence relationships
- Promising developments
  - Paraphrasing relationships between text phrases, tree patterns, DCS trees, etc.
  - Development of resources (e.g. KBs) and shallow semantic parsers to understand semantics
  - Self-improving systems

Integrating & Transforming Natural Language Data - Review

- Covered
  - Transformations on text
  - Lose and tight integration

- More work on
  - Lose integration
  - Optimizing query plans

Integrating & Transforming Natural Language Data – Challenges & Opportunities

- Challenges
  - Lack of schema, opacity of references, richness of semantics and correctness of data
- Much to inspire from
  - Work on transforming text
  - Size and scope of resources for understanding text
  - Progress in shallow semantic parsing
  - Other areas such as translation and speech recognition
- Opportunities
  - Lots of demand for relevant tools
  - More structure in natural language text than text (as a seq. of tokens)
  - Strong ties to deductive databases

* Supported at limited extent

NLIDB: Ideal and Reality

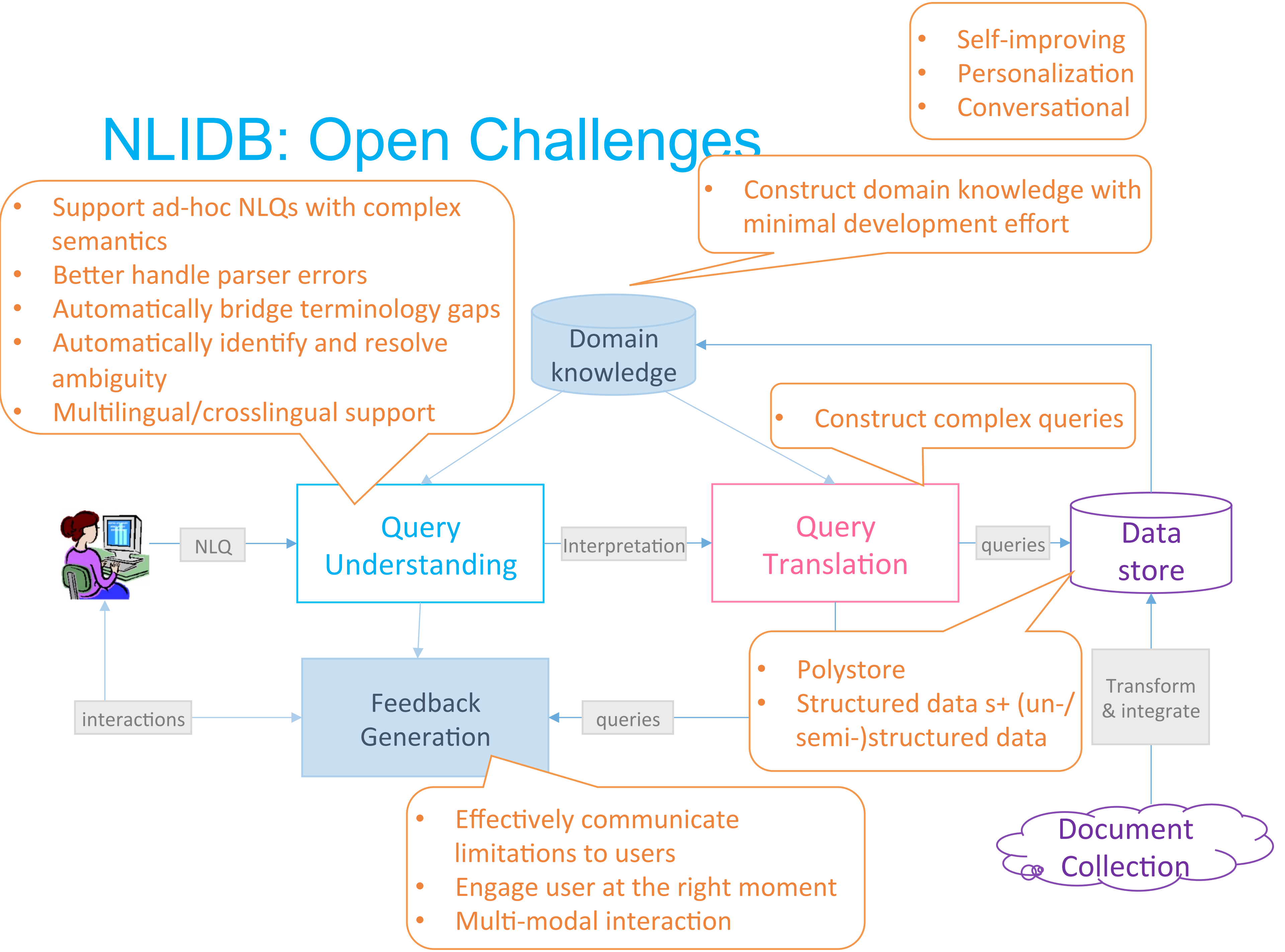
Systems	Scope of NLQ Support		Capability		State		Parsing Error Handling	
	Controlled	Ad-hoc	Fixed	Self-improving	Stateless	Stateful	Auto-correction	Interactive-correction
Ideal NLIDB		✓		✓		✓	✓	

* Supported at limited extent

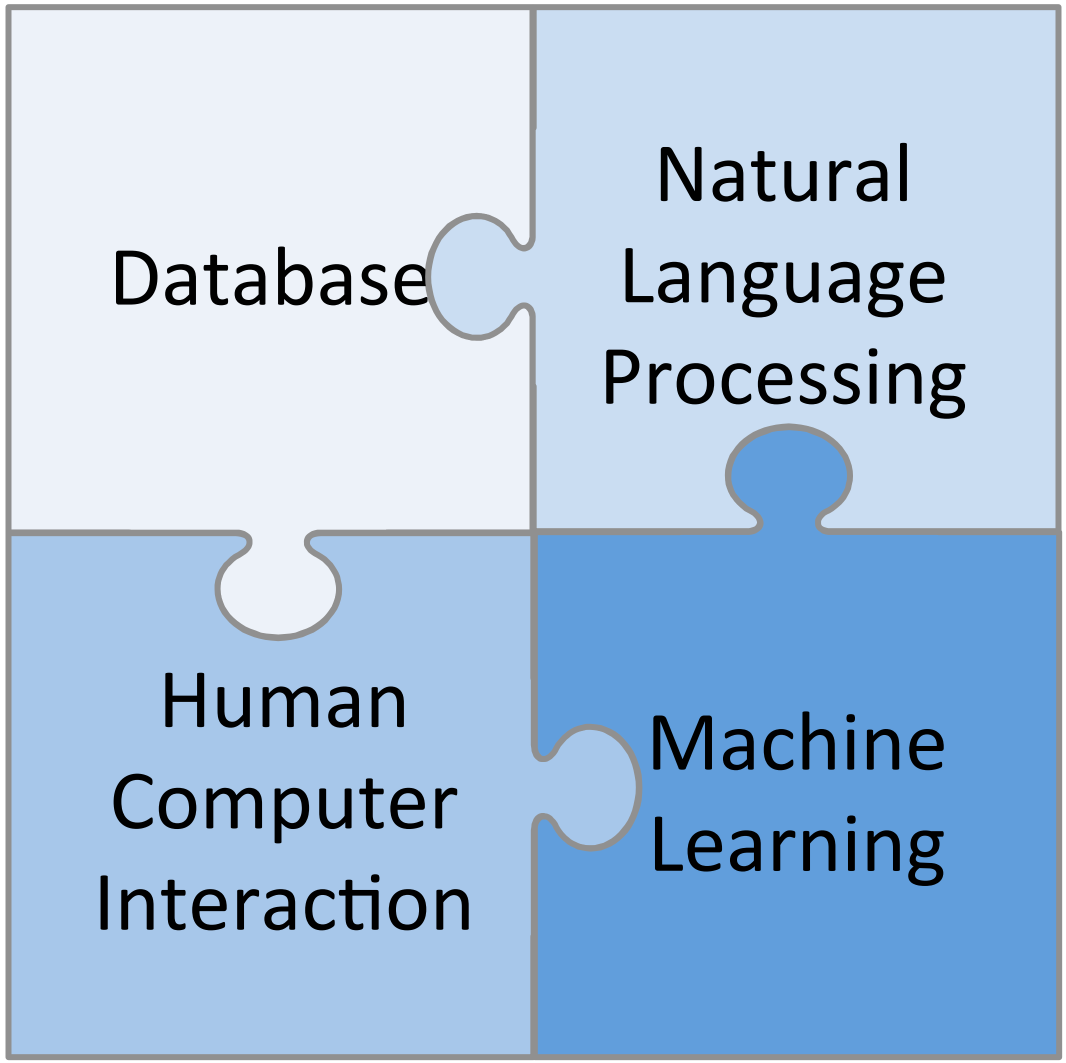
NLIDB: Ideal and Reality – Cont.

Systems	Ambiguity Handling		Query Construction		Target Language
	Automatic	Interactive	Rule-based	Machine-learning	
Ideal NLIDB	✓		✓	✓	Polystore language

NLIDB: Open Challenges



Natural Language DM & Interfaces: Opportunities



References

- [Agichtein and Gravano, 2003] Agichtein, E. and Gravano, L. (2003). Querying text databases for efficient information extraction. In *Proc. of the ICDE Conference*, pages 113–124, Bangalore, India.
- [Agrawal et al., 2008] Agrawal, S., Chakrabarti, K., Chaudhuri, S., and Ganti, V. (2008). Scalable ad-hoc entity extraction from text collections. *PVLDB*, 1(1):945–957.
- [Amsterdamer et al., 2015] Amsterdamer, Y., Kukliansky, A., and Milo, T. (2015). A natural language interface for querying general and individual knowledge. *PVLDB*, 8(12):1430–1441.
- [Andor et al., 2016] Andor, D., Alberti, C., Weiss, D., Severyn, A., Presta, A., Ganchev, K., Petrov, S., and Collins, M. (2016). Globally normalized transition-based neural networks. *CoRR*, abs/1603.06042.
- [Berant et al., 2013] Berant, J., Chou, A., Frostig, R., and Liang, P. (2013). Semantic parsing on freebase from question-answer pairs. In *Proc. of the EMNLP Conference*, volume 2, page 6.
- [Bertino et al., 2012] Bertino, E., Ooi, B. C., Sacks-Davis, R., Tan, K.-L., Zobel, J., Shidlovsky, B., and Andronico, D. (2012). *Indexing techniques for advanced database systems*, volume 8. Springer Science & Business Media.
- [Broder et al., 2003] Broder, A. Z., Carmel, D., Herscovici, M., Soffer, A., and Zien, J. (2003). Efficient query evaluation using a two-level retrieval process. In *Proc. of the CIKM Conf.*, pages 426–434. ACM.
- [Cafarella and Etzioni, 2005] Cafarella, M. J. and Etzioni, O. (2005). A search engine for natural language applications. In *Proc. of the WWW conference*, pages 442–452. ACM.
- [Cafarella et al., 2007] Cafarella, M. J., Re, C., Suciu, D., and Etzioni, O. (2007). Structured querying of web text data: A technical challenge. In *Proc. of the CIDR Conference*, pages 225–234, Asilomar, CA.
- [Cai et al., 2005]Cai, G., Wang, H., MacEachren, A. M., Tokensregex: Defining cascaded regular expressions over tokens. Technical Report CSTR-2014-02, Department of Computer Science, Stanford University.

References – Cont.

- [Chaudhuri et al., 1995] Chaudhuri, S., Dayal, U., and Yan, T. W. (1995). Join queries with external text sources: Execution and optimization techniques. In *ACM SIGMOD Record*, pages 410–422, San Jose, California.
- [Chaudhuri et al., 2004] Chaudhuri, S., Ganti, V., and Gravano, L. (2004). Selectivity estimation for string predicates: Overcoming the underestimation problem. In *Proc. of the ICDE Conf.*, pages 227–238. IEEE.
- [Chen et al., 2000] Chen, Z., Koudas, N., Korn, F., and Muthukrishnan, S. (2000). Selectively estimation for boolean queries. In *Proc. of the PODS Conf.*, pages 216–225. ACM.
- [Chu et al., 2007] Chu, E., Baid, A., Chen, T., Doan, A., and Naughton, J. (2007a). A relational approach to incrementally extracting and querying structure in unstructured data. In *Proc. of the VLDB Conference*.
- [Chubak and Rafiei, 2010] Chubak, P. and Rafiei, D. (2010). Index Structures for Efficiently Searching Natural Language Text. In *Proc. of the CIKM Conference*.
- [Chubak and Rafiei, 2012] Chubak, P. and Rafiei, D. (2012). Efficient indexing and querying over syntactically annotated trees. *PVLDB*, 5(11):1316–1327.
- [Codd, 1974] Codd, E. (1974). Seven steps to rendezvous with the casual user. In *IFIP Working Conference Data Base Management*, pages 179–200.
- [Ferrucci, 2012] Ferrucci, D. A. (2012). Introduction to ”this is watson”. *IBM Journal of Research and Development*, 56(3):1.
- [Gonnet and Tompa, 1987] Gonnet, G. H. and Tompa, F. W. (1987). Mind your grammar: a new approach to modelling text. In *Proc. of the VLDB Conference*, pages 339–346, Brighton, England.
- [Gyssens et al., 1989] Gyssens, M., Paredaens, J., and Gucht, D. V. (1989). A grammar-based approach towards unifying hierarchical data models (extended abstract). In *Proc. of the SIGMOD Conference*, pages 263–272, Portland, Oregon.

References – Cont.

- [Jagadish et al., 1999] Jagadish, H., Ng, R. T., and Srivastava, D. (1999). Substring selectivity estimation. In *Proc. of the PODS Conf.*, pages 249–260. ACM.
- [Jain et al., 2008] Jain, A., Doan, A., and Gravano, L. (2008). Optimizing SQL queries over text databases. In *Proc. of the ICDE Conference*, pages 636–645, Cancun, Mexico.
- [Kaoudi and Manolescu, 2015] Kaoudi, Z. and Manolescu, I. (2015). Rdf in the clouds: a survey. *The VLDB Journal*, 24(1):67–91.
- [Lewis and Steedman, 2013] Lewis, M. and Steedman, M. (2013). Combining distributional and logical semantics. *Transactions of the Association for Computational Linguistics*, 1:179–192.
- [Li and Jagadish, 2014] Li, F. and Jagadish, H. V. (2014). Constructing an interactive natural language interface for relational databases. *PVLDB*, 8(1):73–84.
- [Li et al., 2007] Li, Y., Yang, H., and Jagadish, H. V. (2007). Nalix: A generic natural language search environment for XML data. *ACM Trans. Database Systems*, 32(4).
- [Liang et al., 2011] Liang, P., Jordan, M. I., and Klein, D. (2011). Learning dependency-based compositional semantics. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 590–599. Association for Computational Linguistics.
- [Lin and Pantel, 2001] Lin, D. and Pantel, P. (2001). Dirt - discovery of inference rules from text. In *Proc. of the KDD Conference*, pages 323–328.
- [Rafiei and Li, 2009] Rafiei, D. and Li, H. (2009). Data extraction from the web using wild card queries. In *Proc. of the CIKM Conference*, pages 1939–1942.
- [Ravichandran and Hovy, 2002] Ravichandran, D. and Hovy, E. (2002). Learning surface text patterns for a question answering system. In *Proc. of the ACL Conference*.

References – Cont.

- [Popescu et al., 2004] Popescu et al., A. (2004). Modern natural language interfaces to databases: Composing statistical parsing with semantic tractability. In *Proc. of the COLING Conference*.
- [Saha et al., 2016] Saha, D., Floratou, A., Sankaranarayanan, K., Minhas, U. F., Mittal, A. R., and O¨ zcan, F. (2016). Athena: An ontology-driven system for natural language querying over relational data stores. *PVLDB*, 9(12):1209–1220.
- [Salminen and Tompa, 1994] Salminen, A. and Tompa, F. (1994). PAT expressions: an algebra for text search.
- *Acta Linguistica Hungarica*, 41(1):277–306.
- [Stratica et al., 2005] Stratica, N., Kosseim, L., and Desai,
- B. C. (2005). Using semantic templates for a natural language interface to the cindi virtual library. *Data and Knowledge Engineering*, 55(1):4–19.
- [Suchanek and Preda, 2014] Suchanek, F. M. and Preda,
- N. (2014). Semantic culturomics. *Proc. of the VLDB Endowment*, 7(12):1215–1218.
- [Tague et al., 1991] Tague, J., Salminen, A., and McClellan, C. (1991). A complete model for information retrieval systems. In *Proc. of the SIGIR Conference*, pages 14–20, Chicago, Illinois.
- [Tian et al., 2014] Tian, R., Miyao, Y., and Matsuzaki, T. (2014). Logical inference on dependency-based compositional semantics. In *Proc. of the ACL Conference*, pages 79–89.
- [Wu and Palmer, 1994] Wu, Z. and Palmer, M. (1994). Verbs semantics and lexical selection. In ACL
- [Valenzuela-Escarcega et al., 2016] Valenzuela-Escarcega, M. A., Hahn-Powell, G., and Surdeanu, M. (2016). Odin's runes: A rule language for information extraction. In *Proc. of the Language Resources and Evaluation Conference (LREC)*.
- [Xu, 2014] Xu, W. (2014). *Data-driven approaches for paraphrasing across language variations*. PhD thesis, New York University.

Relevant Tutorials

- Semantic parsing
  - Percy Liang: “natural language understanding: foundations and state-of-the-art”, ICML 2015.

- Information extraction
  - Laura Chiticariu, Yunyao Li, Sriram Raghavan, Frederick Reiss: “Enterprise information extraction: recent developments and open challenges.” SIGMOD 2010

- Entity resolution
  - Lise Getoor and shwin Machanavajjhala: “Entity Resolution for Big Data” KDD 2013