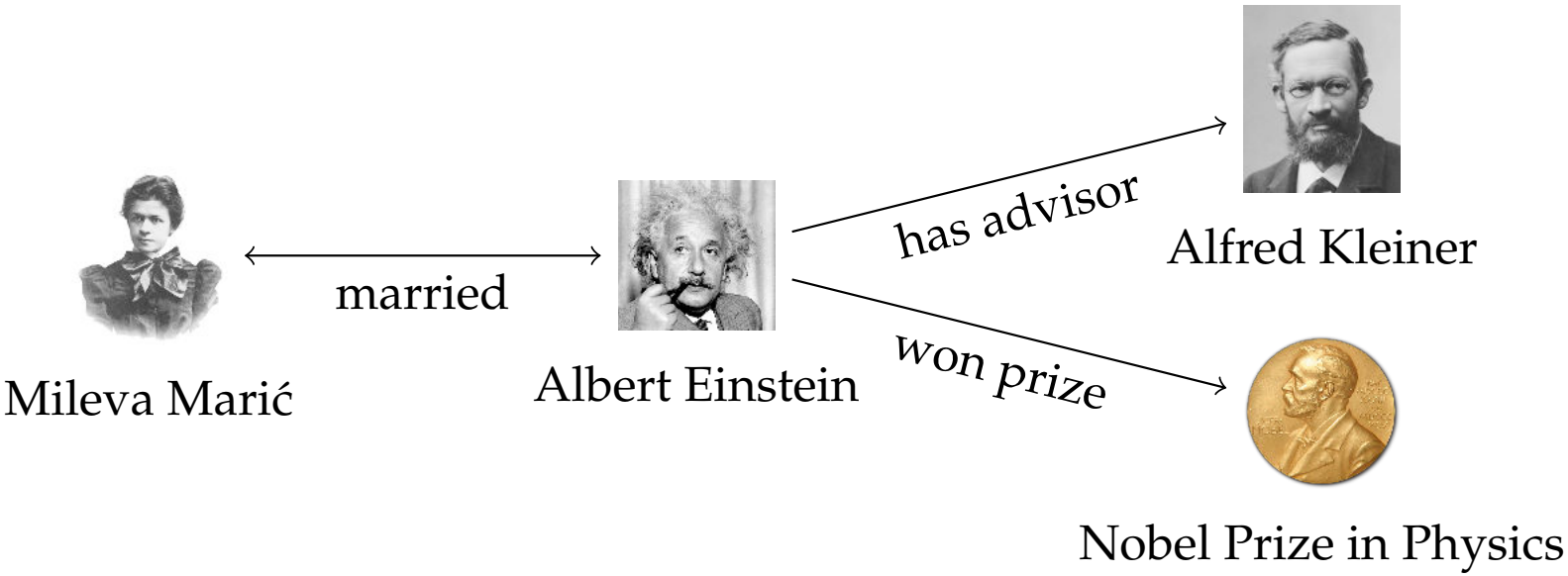
Expanding the YAGO knowledge base

Thomas Rebele

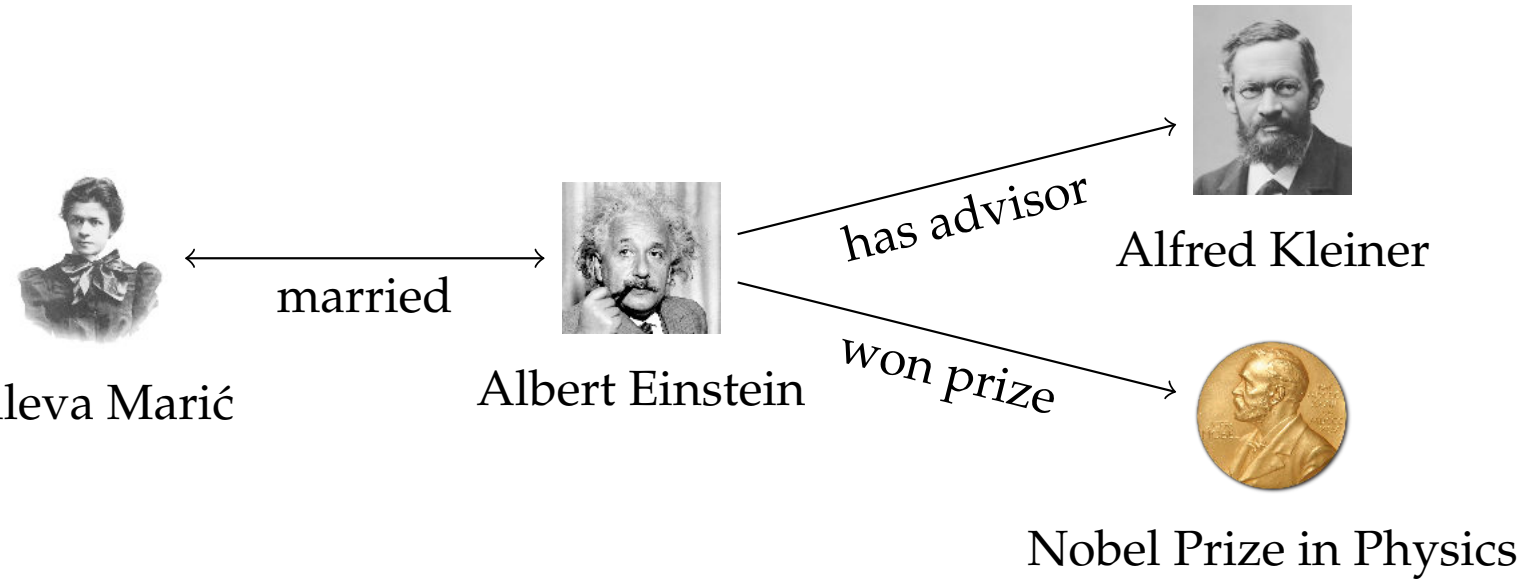
Télécom ParisTech

2018-07-19

What is a knowledge base?



What is a knowledge base?



Applications of knowledge bases:

- Question answering
- Semantic search
- Text analysis
- Machine translation

What is YAGO?

The YAGO knowledge base
What is a knowledge base?
What is YAGO?
Involvement
Outline
Using YAGO for the humanities
Adding Words to Regexes
Answering Queries with Unix Shell
Conclusion

- Knowledge base with 10 million entities and >210 million facts
- Automatically extracted from Wikipedia, Wordnet, and Geonames
- Multilingual facts from 10 languages
- Focus on precision
- Developed by Max-Planck Institute for Informatics and Télécom ParisTech

yago
select knowledge

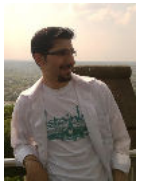
TELECOM ParisTech

max planck institut informatik







What is YAGO?

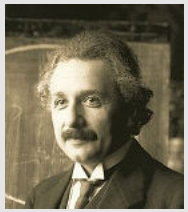
Languages

English
German
French

Albert Einstein

Albert Einstein was a physicist. His work influenced the philosophy of science. He developed the theory of relativity.

Albert Einstein



Spouse(s)	Mileva Marić
Doctoral advisor(s)	Alfred Kleiner

Categories: Nobel laureates in Physics

automatic extraction

Mileva Marić

married

Albert Einstein

has advisor

Alfred Kleiner

won prize

Nobel Prize in Physics

Involvement

- I joined the project in 2014
- Maintenance and development
- Contributed to open source release in 2017 at `https://github.com/yago-naga/yago3/`
- Coordinated / contributed to the evaluation
  - ground truth: Wikipedia
  - 98% facts of the sample were correct

Publication: ISWC 2016 (resource paper)


Thomas Rebele


Fabian Suchanek


Johannes Hoffart


Joanna Biega


Erdal Kuzey

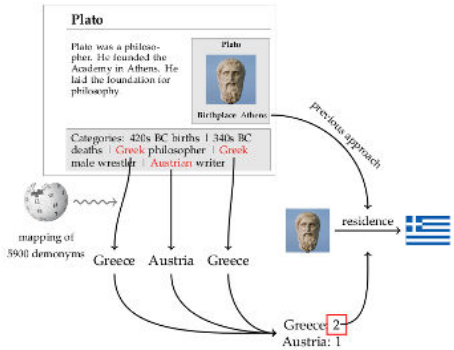

Gerhard Weikum

YAGO is very accurate. But how complete is it?

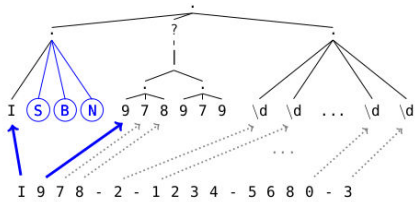
Expanding the YAGO knowledge base
Rebele
The YAGO knowledge base
What is a knowledge base?
What is YAGO?
Involvement
Outline
Using YAGO for the humanities
Adding Words to Regexes
Answering Queries with Unix Shell
Conclusion

Contributions:

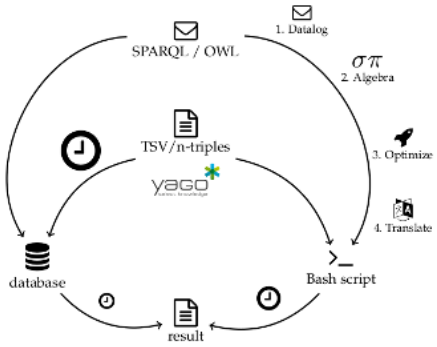
Extracting more information about residences, gender, birth and death dates



Repairing regular expressions by adding missing words

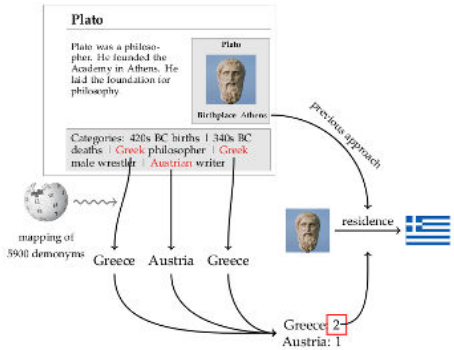


Preprocessing tabular data by transforming queries to Bash scripts

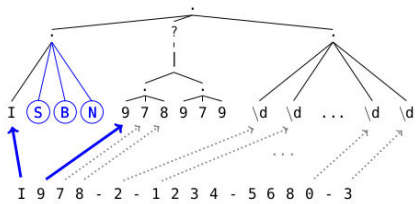


Contributions:

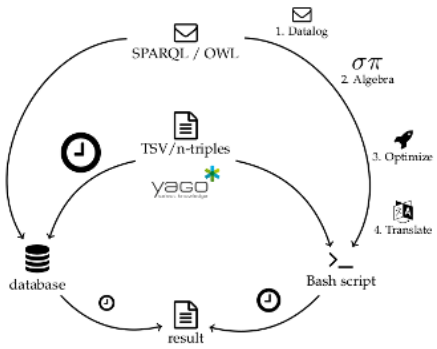
Extracting more information about residences, gender, birth and death dates



Repairing regular expressions
by adding missing words



Preprocessing tabular data
by transforming queries to Bash scripts



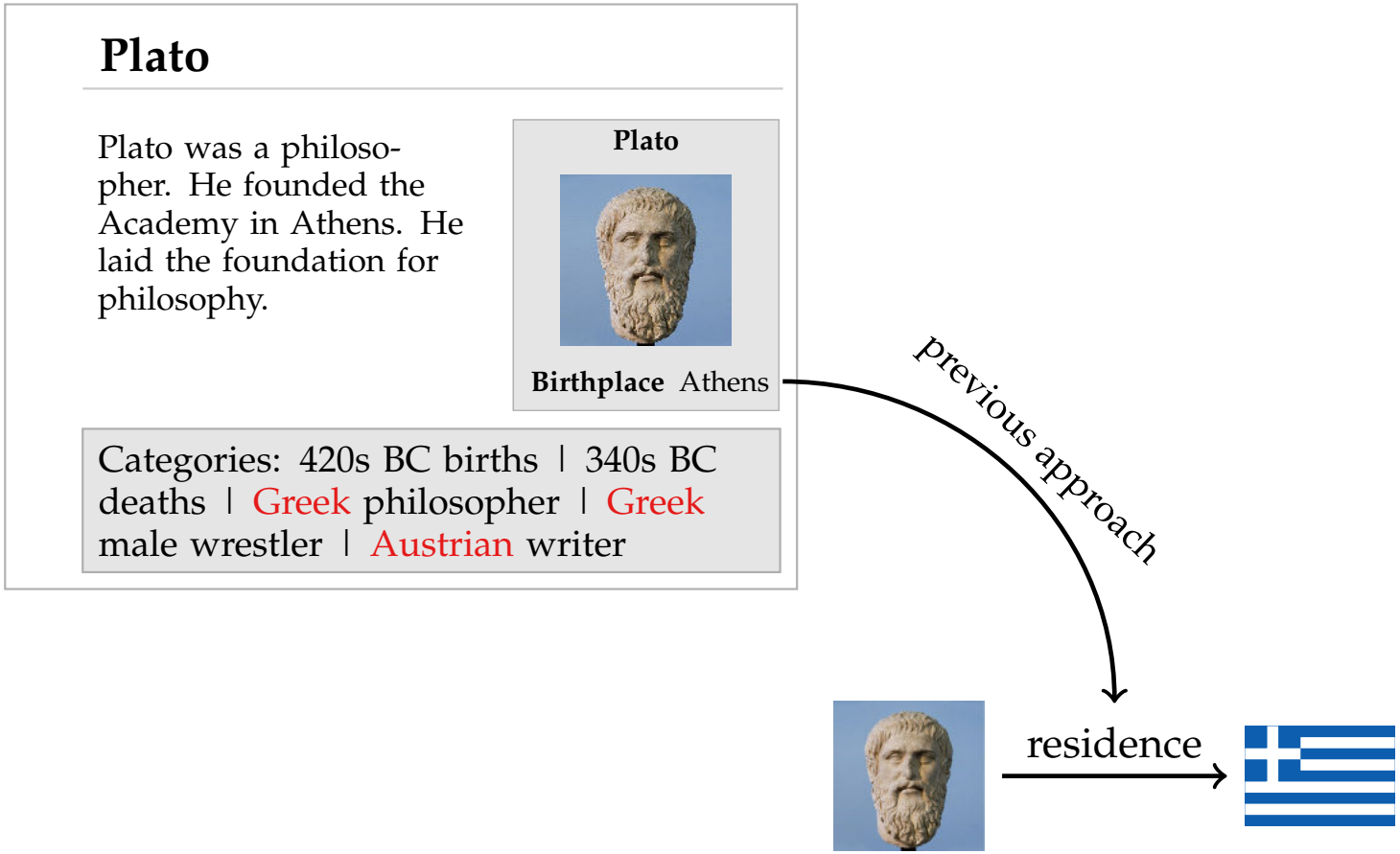
Using YAGO for the humanities: Problem statement

- Every person lives somewhere,
  but YAGO knows the residence only for 30% of the people
- Every person has a gender,
  but YAGO knows the gender only for 64% of the people

How can we make YAGO more complete?

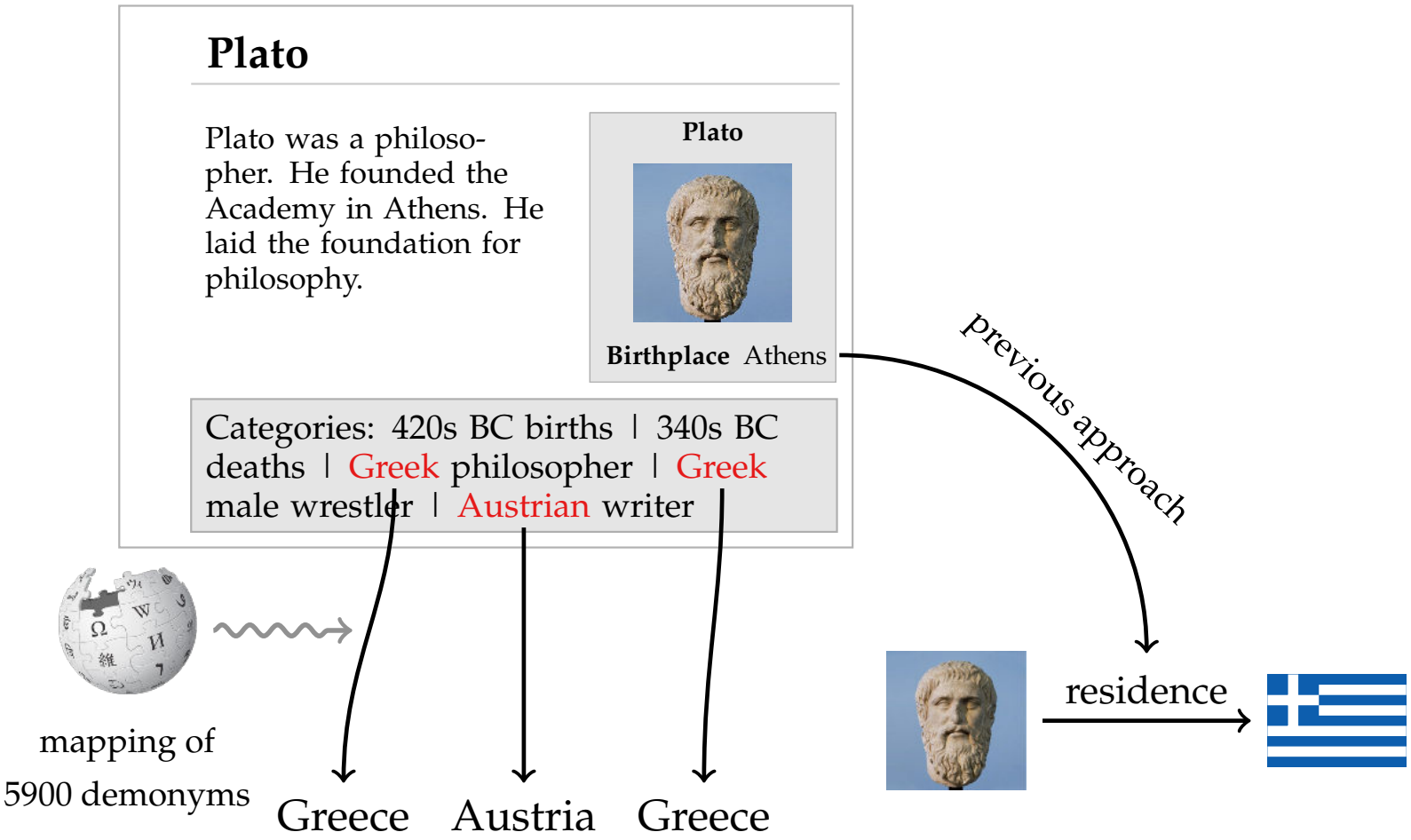
Expanding the YAGO knowledge base
Rebele
The YAGO knowledge base
Outline
Using YAGO for the humanities
Problem statement
Extensions
Place of residence
Gender
Evaluation
Births per month
Life span over time
Relative population size
Summary
Adding Words to Regexes
Answering Queries with Unix Shell
Conclusion

Using YAGO for the humanities: Place of residence



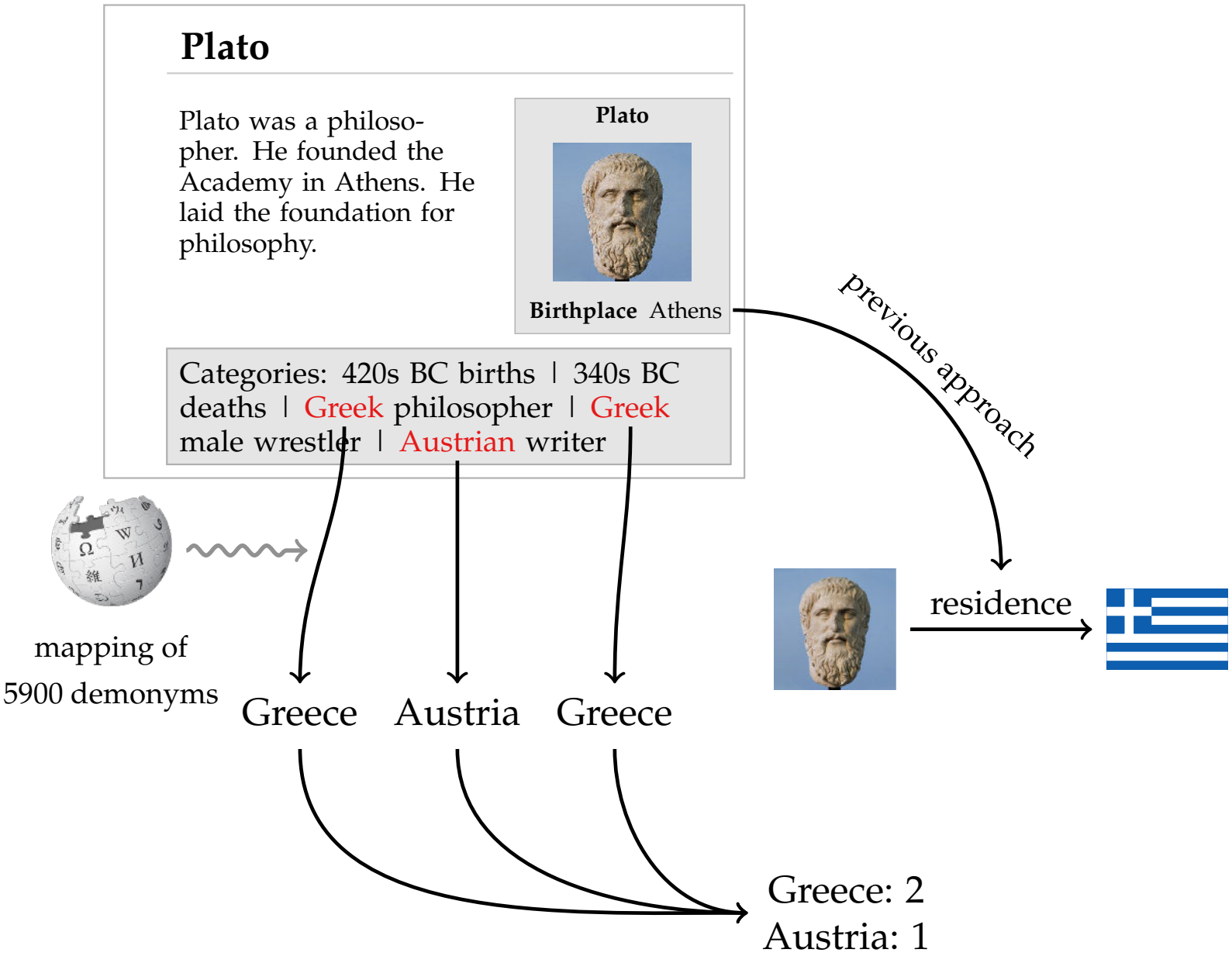
Expanding the YAGO knowledge base
Rebele
The YAGO knowledge base
Outline
Using YAGO for the humanities
Problem statement
Extensions
Place of residence
Gender
Evaluation
Births per month
Life span over time
Relative population size
Summary
Adding Words to Regexes
Answering Queries with Unix Shell
Conclusion

Using YAGO for the humanities: Place of residence

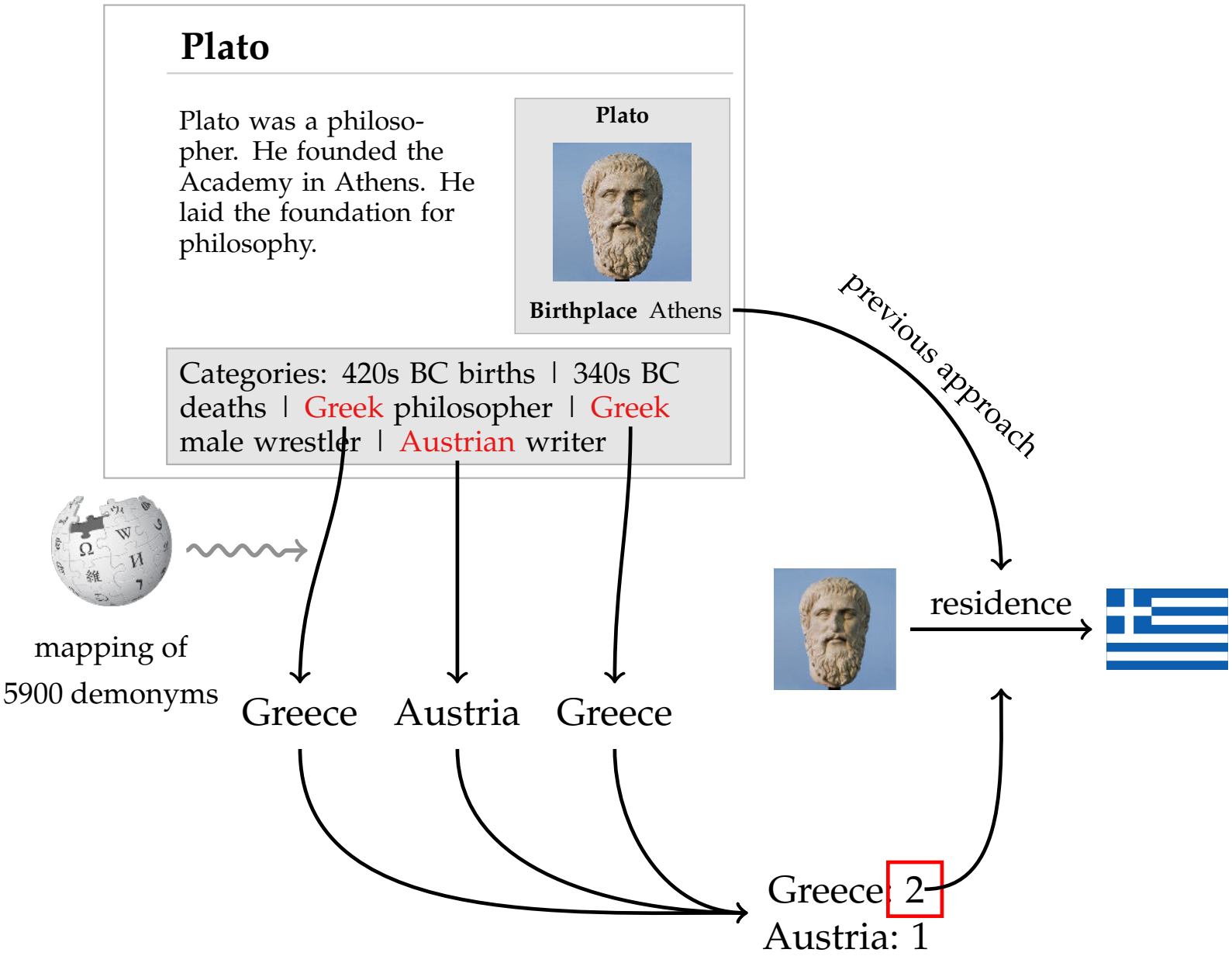


Expanding the YAGO knowledge base
Rebele
The YAGO knowledge base
Outline
Using YAGO for the humanities
Problem statement
Extensions
Place of residence
Gender
Evaluation
Births per month
Life span over time
Relative population size
Summary
Adding Words to Regexes
Answering Queries with Unix Shell
Conclusion

Using YAGO for the humanities: Place of residence

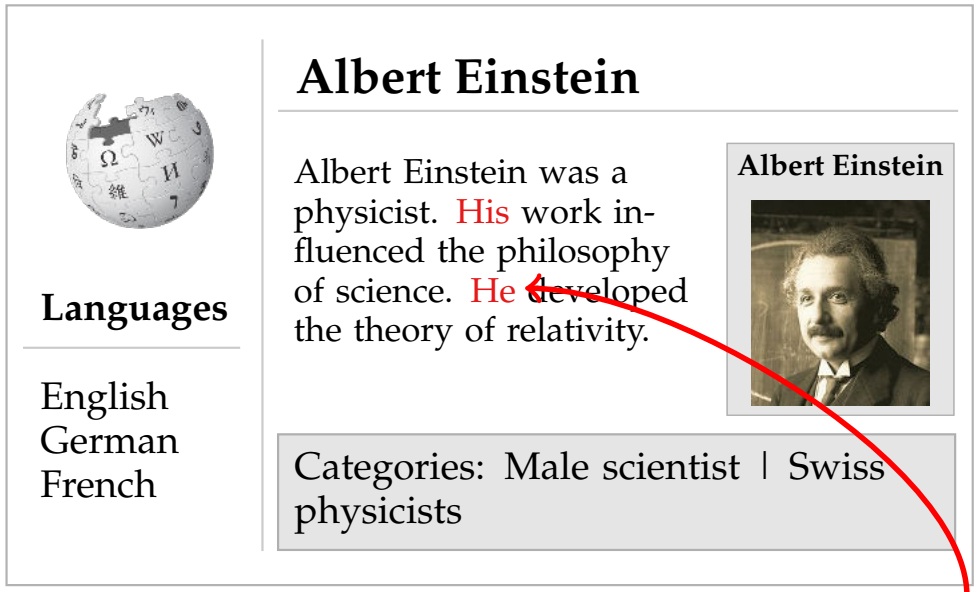


Using YAGO for the humanities: Place of residence



Using YAGO for the humanities: Gender

Extract gender:



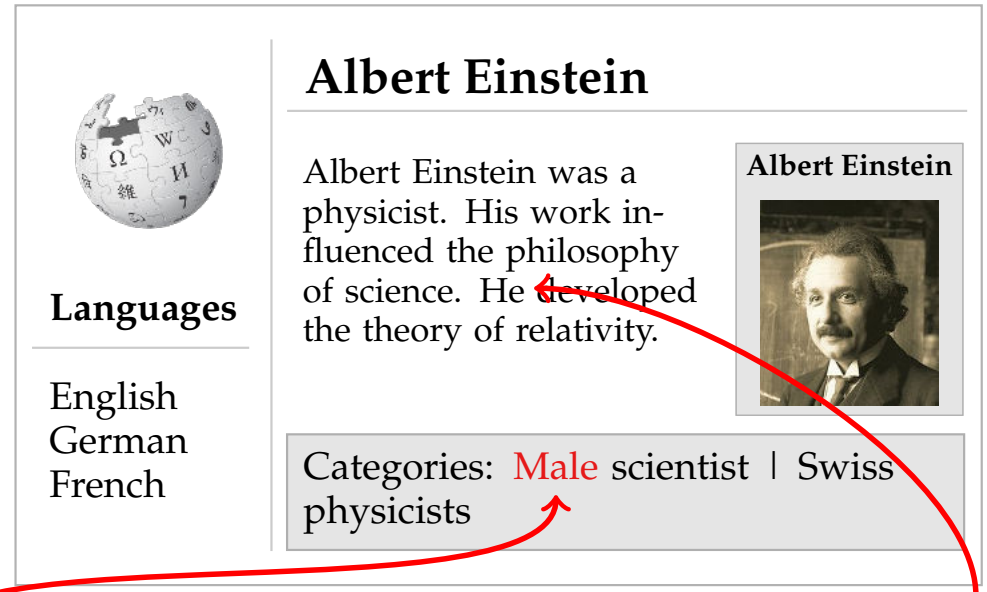
From pronoun:

- YAGO's original algorithm
- Count pronouns (he, him / she, her)
- Assign gender accordingly

The YAGO knowledge base
Outline
Using YAGO for the humanities
Problem statement
Extensions
Place of residence
Gender
Evaluation
Births per month
Life span over time
Relative population size
Summary
Adding Words to Regexes
Answering Queries with Unix Shell
Conclusion

Using YAGO for the humanities: Gender

Extract gender:



From category

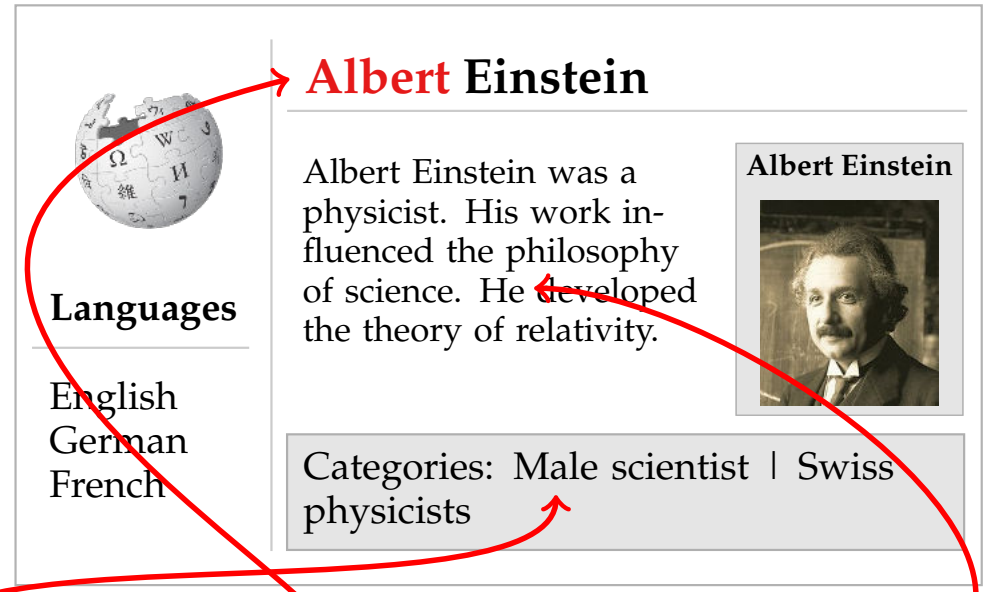
From pronoun:

- YAGO's original algorithm
- Count pronouns (he, him / she, her)
- Assign gender accordingly

The YAGO knowledge base
Outline
Using YAGO for the humanities
Problem statement
Extensions
Place of residence
Gender
Evaluation
Births per month
Life span over time
Relative population size
Summary
Adding Words to Regexes
Answering Queries with Unix Shell
Conclusion

Using YAGO for the humanities: Gender

Extract gender:



From category

From first name:

- Count males/females for each first name
- Assign names to gender accordingly

From pronoun:

- YAGO's original algorithm
- Count pronouns (he, him / she, her)
- Assign gender accordingly

The YAGO knowledge base
Outline
Using YAGO for the humanities
Problem statement
Extensions
Place of residence
Gender
Evaluation
Births per month
Life span over time
Relative population size
Summary
Adding Words to Regexes
Answering Queries with Unix Shell
Conclusion

Using YAGO for the humanities: Evaluation

- Compare extraction process on Wikipedia dump from 2017-02-20
- Extracted on 11 languages
- Evaluate precision based on a sample of 100 people

Extraction	YAGO before	Recall	YAGO now	Recall	Precision	DBpedia (en)
Place of residence	0.7m	30%	2.1m	91% (+201%)	97% (*)	0.7m
Gender	1.5m	64%	2.0m	87% (+35%)	98%	4k
Birth dates	1.6m	69%	1.7m	74% (+8%)	100%	0.8m
Death dates	0.7m	33%	0.8m	36% (+10%)	100%	0.3m

Table: Coverage and precision of our methods.
Recall relative to total number of people in YAGO (2.2m).

m million k thousand

(*) 6% of anachronistic residencies (e.g., German Empire instead of Germany)

Using YAGO for the humanities: Births per month

The YAGO knowledge base

Outline

Using YAGO for the humanities

Problem statement

Extensions

Place of residence

Gender

Evaluation

Births per month

Life span over time

Relative population size

Summary

Adding Words to Regexes

Answering Queries with Unix Shell

Conclusion

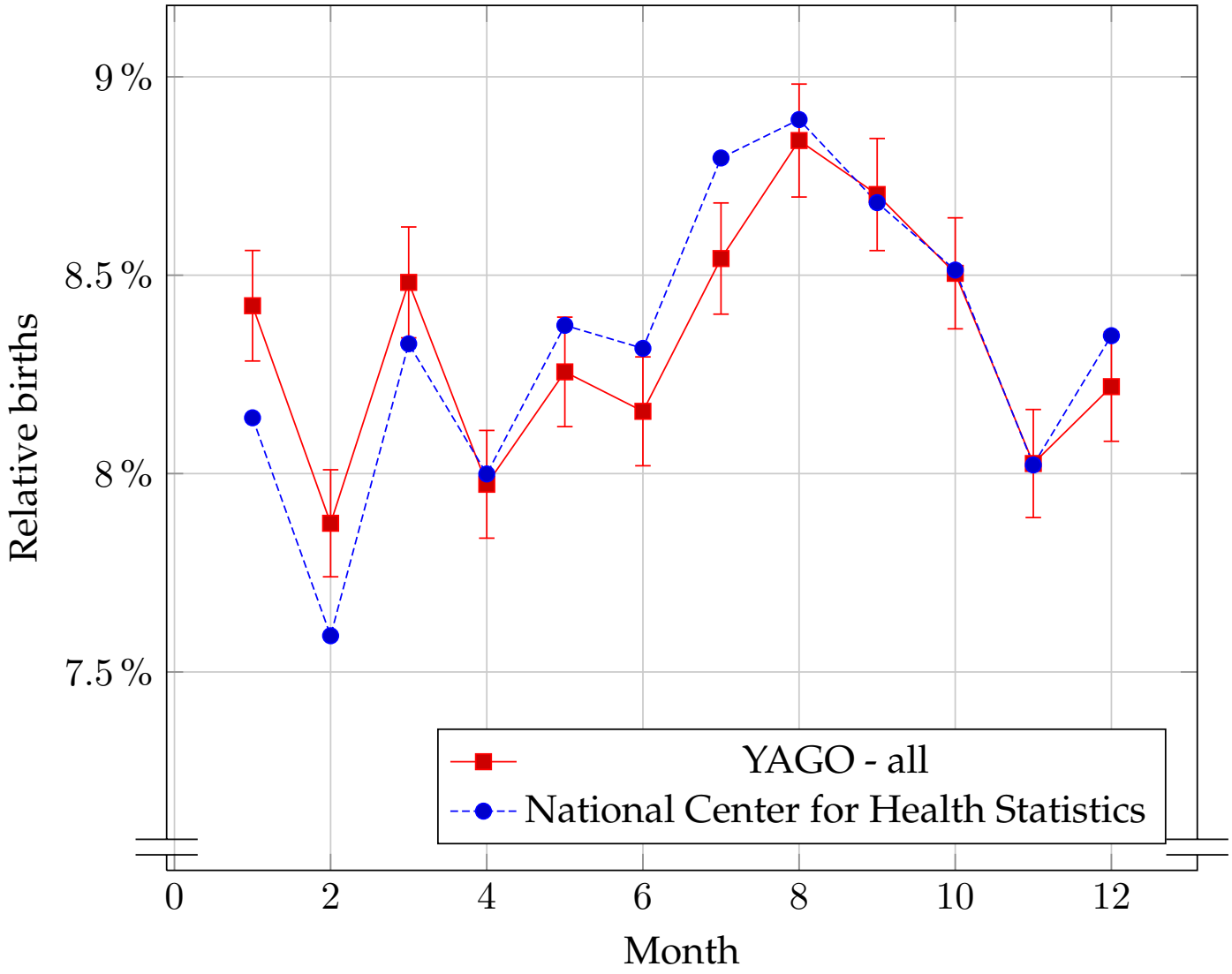


Figure: Births per month in the United States between 2003 and 2015 (with the Student's t confidence interval at $\alpha = 95\%$).

Using YAGO for the humanities: Births per month

Expanding the YAGO knowledge base

Rebele

The YAGO knowledge base

Outline

Using YAGO for the humanities

Problem statement

Extensions

Place of residence

Gender

Evaluation

Births per month

Life span over time

Relative population size

Summary

s to

Unix

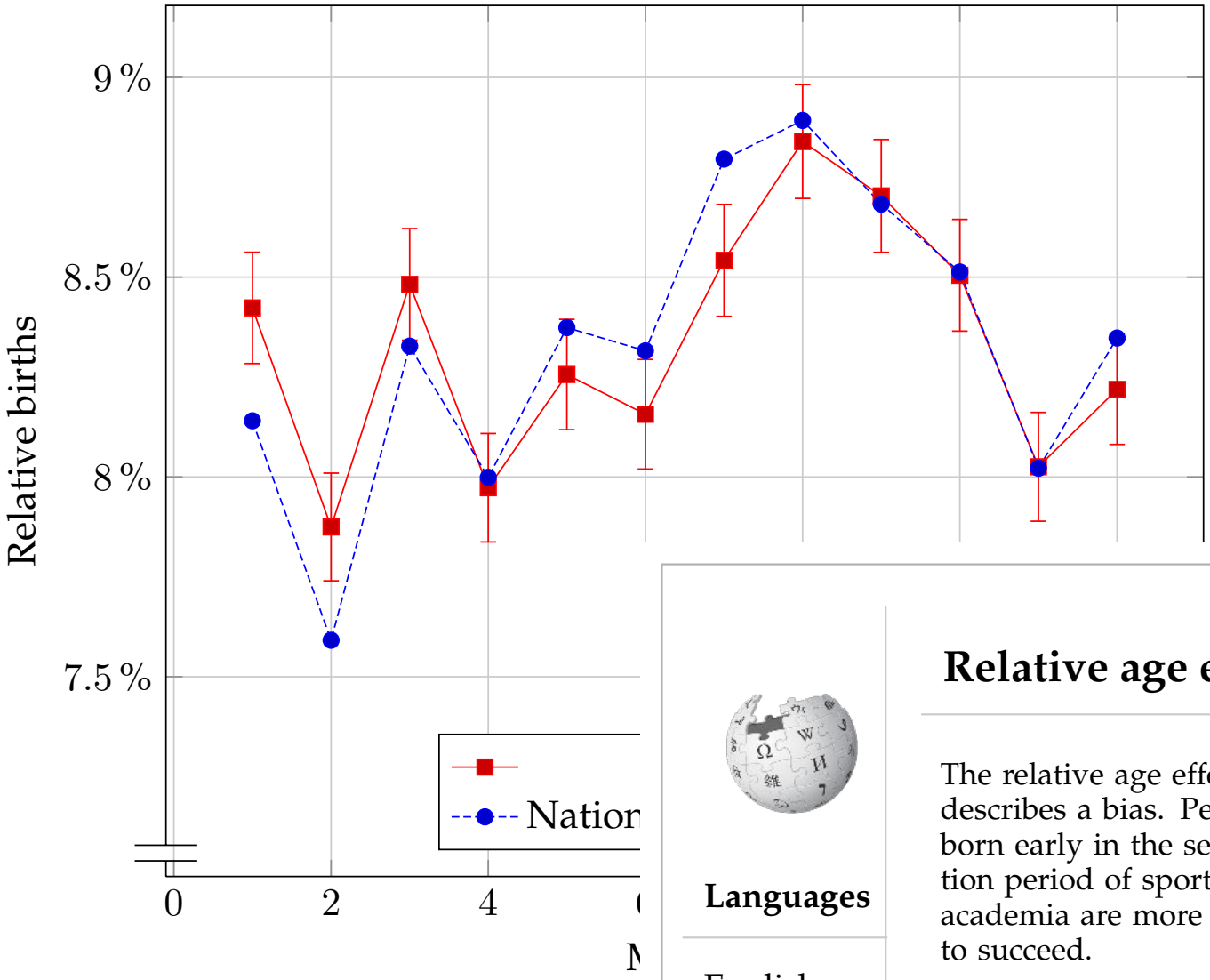


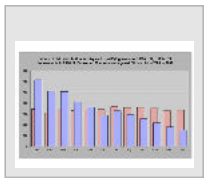
Figure: Births per month in the United Sta
(with the Student's t confidence interval a

Languages

English
Euskara

Relative age effect

The relative age effect describes a bias. People born early in the selection period of sports or academia are more likely to succeed.



Categories: Ageism | Epidemiology

Using YAGO for the humanities: Births per month

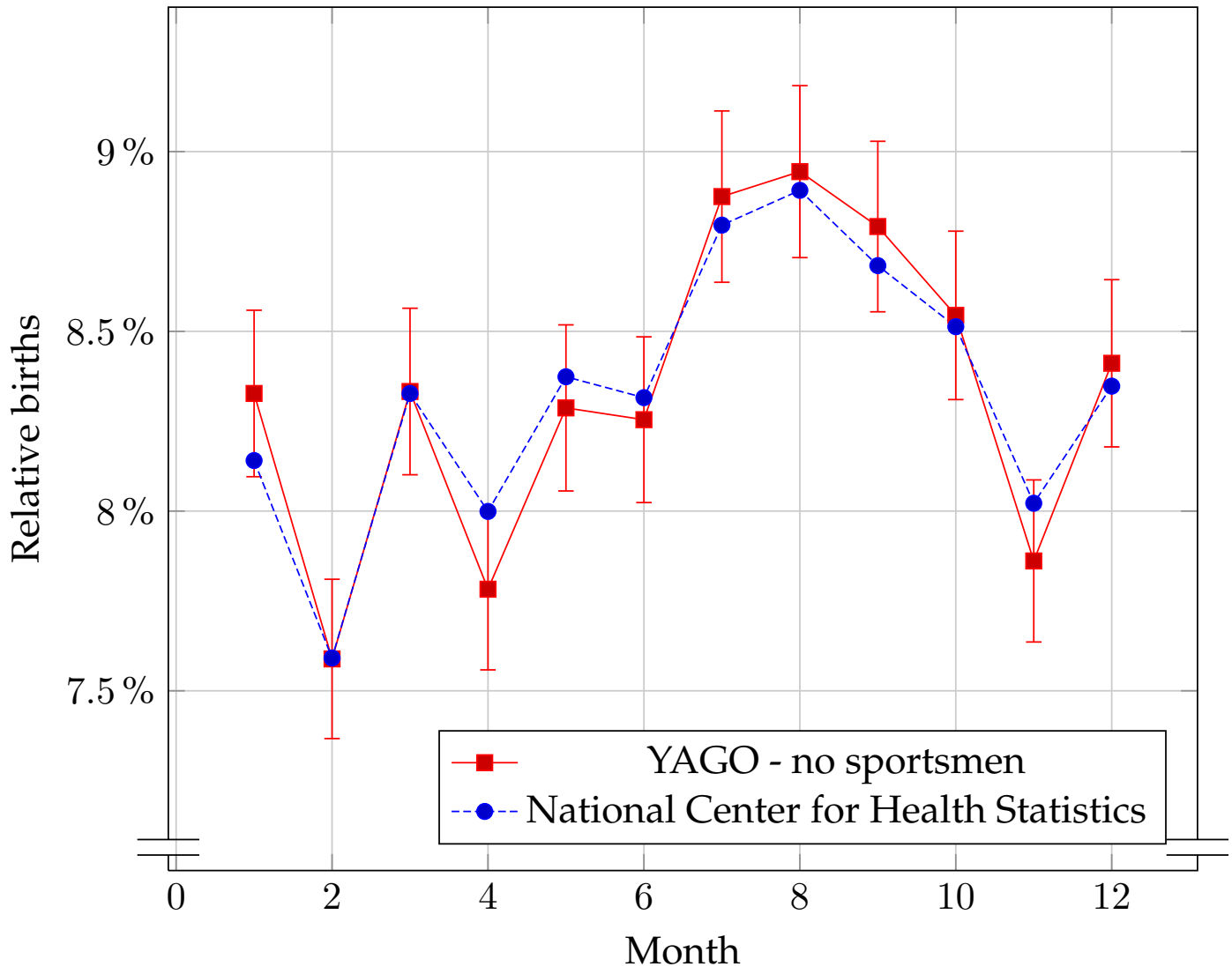


Figure: Births per month in the United States between 2003 and 2015 (with the Student's t confidence interval at $\alpha = 95\%$).

Expanding the YAGO knowledge base

Rebele

The YAGO knowledge base

Outline

Using YAGO for the humanities

Problem statement

Extensions

Place of residence

Gender

Evaluation

Births per month

Life span over time

Relative population size

Summary

Adding Words to Regexes

Answering Queries with Unix Shell

Conclusion

Using YAGO for the humanities: Life span over time

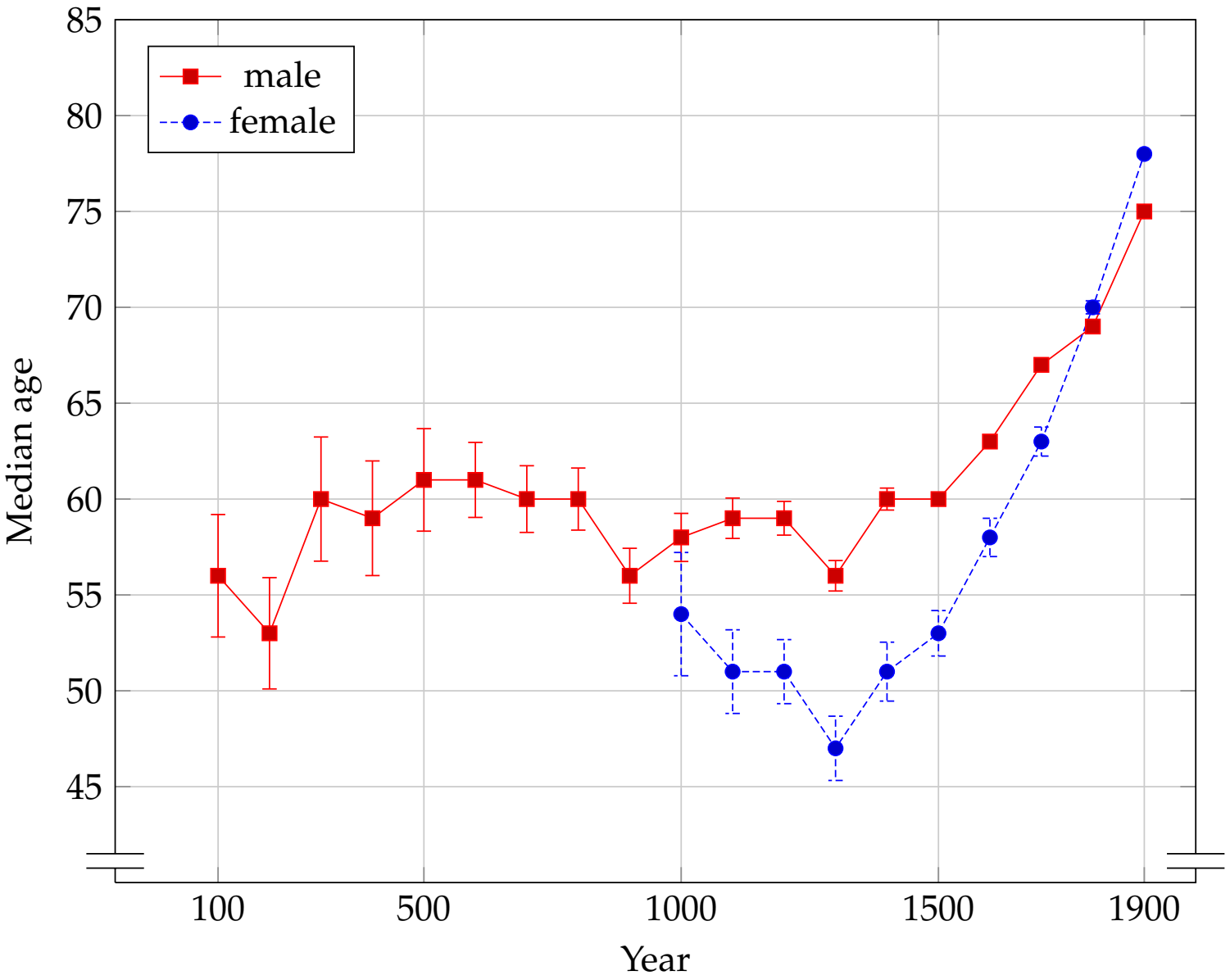


Figure: Median age over time, by year of birth

Expanding the YAGO knowledge base
Rebele
The YAGO knowledge base
Outline
Using YAGO for the humanities
Problem statement
Extensions
Place of residence
Gender
Evaluation
Births per month
Life span over time
Relative population size
Summary
Adding Words to Regexes
Answering Queries with Unix Shell
Conclusion

Using YAGO for the humanities: Relative population size

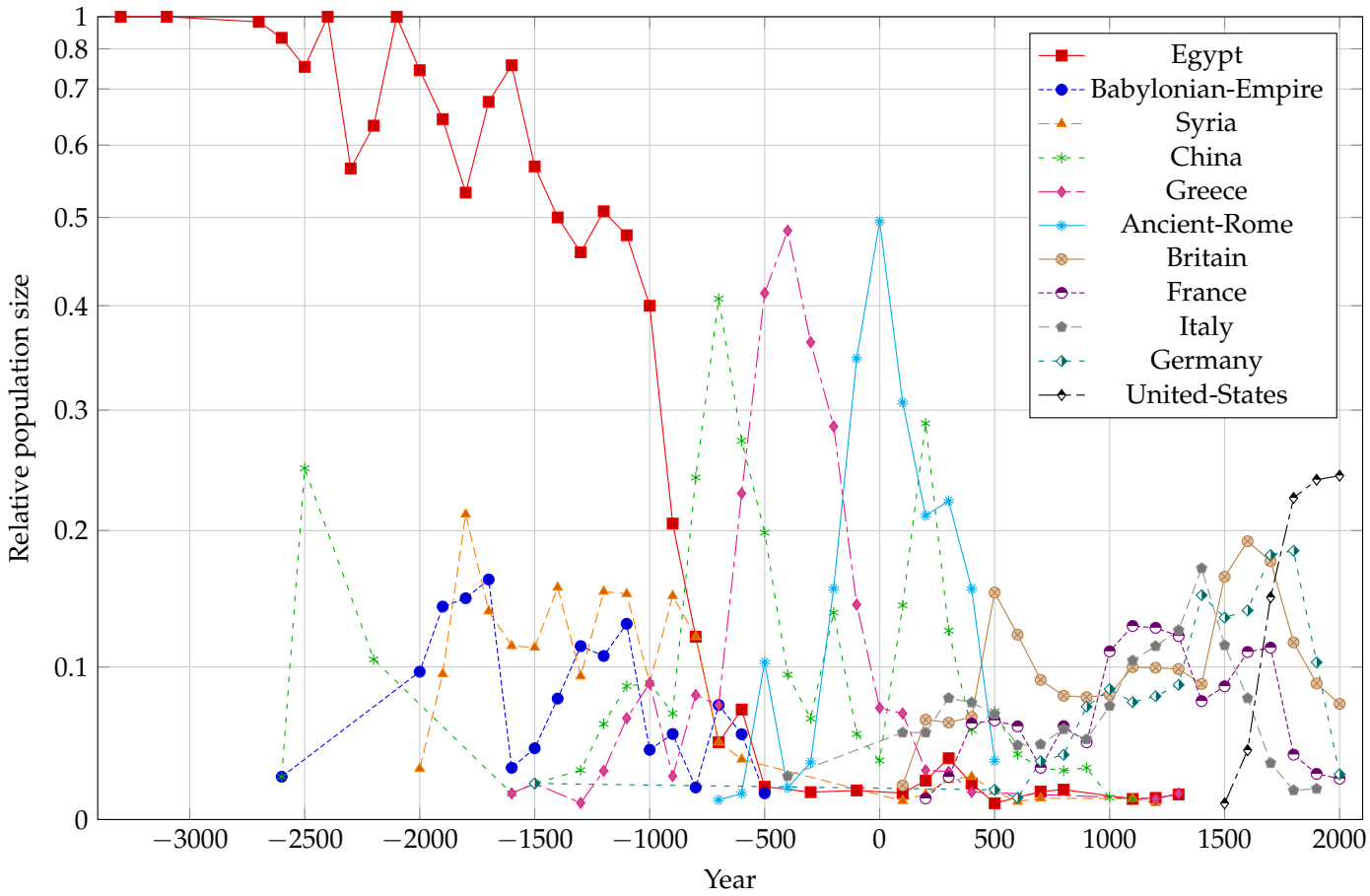


Figure: Relative population size, by century. The y-axis is scaled by a quadratic function.

Expanding the YAGO knowledge base

Rebele

The YAGO knowledge base

Outline

Using YAGO for the humanities

Problem statement

Extensions

Place of residence

Gender

Evaluation

Births per month

Life span over time

Relative population size

Summary

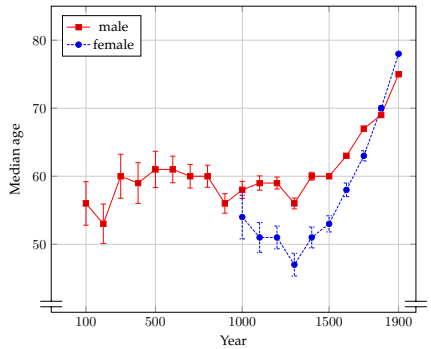
Adding Words to Regexes

Answering Queries with Unix Shell

Conclusion

Using YAGO for the humanities: Summary

- Extension of YAGO:
  - More people with residences (+201%, 97% precison)
  - More people with genders (+35%, 98% precision)
  - More birth and death dates (+8%/10%, 100% precision)
- Case studies:
  - Births per month
  - Life span over time
  - Relative population size over time
- Interdisciplinary project



Publication: ISWC 2017 (workshop paper)


Thomas Rebele


Arash Nekoei


Fabian Suchanek

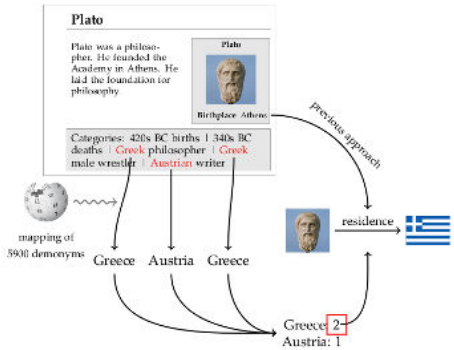
We often had to repair regular expressions (e.g., for matching dates). Can we automate this step?

Expanding the YAGO knowledge base
Rebele
The YAGO knowledge base
Outline
Using YAGO for the humanities
Problem statement
Extensions
Place of residence
Gender
Evaluation
Births per month
Life span over time
Relative population size
Summary
Adding Words to Regexes
Answering Queries with Unix Shell
Conclusion

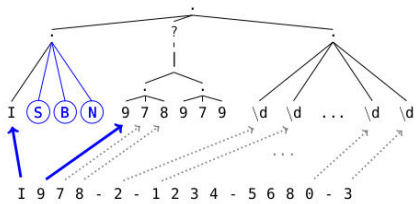
Using YAGO for the humanities: Summary

Contributions:

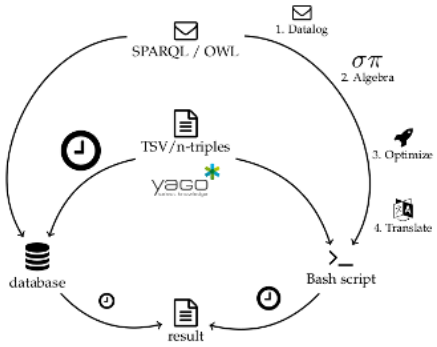
Extracting more information about residences, gender, birth and death dates



Repairing regular expressions by adding missing words

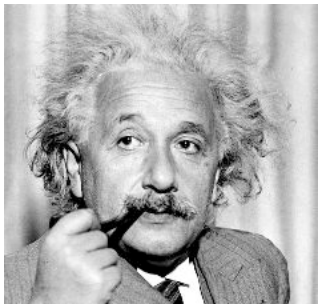


Preprocessing tabular data by transforming queries to Bash scripts



The YAGO knowledge base
Outline
Using YAGO for the humanities
Problem statement
Extensions
Place of residence
Gender
Evaluation
Births per month
Life span over time
Relative population size
Summary
Adding Words to Regexes
Answering Queries with Unix Shell
Conclusion

Adding Words to Regexes: Introduction



Why does YAGO not know
the ISBN numbers of my books?

- We want to find ISBN numbers in Wikipedia to include it in YAGO
- We try the regex `ISBN(978|979)?\d{10}`

Expanding the YAGO knowledge base

Rebele

The YAGO knowledge base

Outline

Using YAGO for the humanities

Adding Words to Regexes

Introduction

Problem statement

What is new in our approach

Approximate regex matching

Finding the gaps

Add missing parts

Feedback function

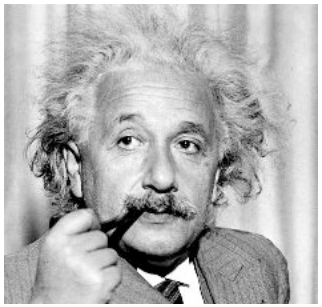
Experiments

Summary

Answering Queries with Unix Shell

Conclusion

Adding Words to Regexes: Introduction



Why does YAGO not know
the ISBN numbers of my books?

- We want to find ISBN numbers in Wikipedia to include it in YAGO
- We try the regex `ISBN(978|979)?\d{10}`
- Why does the regex not find `I978-2-1234-5680-3` ?
- How can we modify the regex automatically to match the word?

Expanding the YAGO knowledge base

Rebele

The YAGO knowledge base

Outline

Using YAGO for the humanities

Adding Words to Regexes

Introduction

Problem statement

What is new in our approach

Approximate regex matching

Finding the gaps

Add missing parts

Feedback function

Experiments

Summary

Answering Queries with Unix Shell

Conclusion

Adding Words to Regexes: Problem statement

The YAGO knowledge base

Outline

Using YAGO for the humanities

Adding Words to Regexes

Introduction

Problem statement

What is new in our approach

Approximate regex matching

Finding the gaps

Add missing parts

Feedback function

Experiments

Summary

Answering Queries with Unix Shell

Conclusion

Problem statement, first try:

Given

- a regular expression r and
- a set of positive examples E^+,

find a regular expression r' such that

- $L(r) \subseteq L(r')$
- $E^+ \subseteq L(r')$

```
ISBN(978|979)?\d{10}
```

{ I978-2-1234-5680-3 }

Adding Words to Regexes: Problem statement

The YAGO knowledge base

Outline

Using YAGO for the humanities

Adding Words to Regexes

Introduction

Problem statement

What is new in our approach

Approximate regex matching

Finding the gaps

Add missing parts

Feedback function

Experiments

Summary

Answering Queries with Unix Shell

Conclusion

Problem statement, first try:

Given

- a regular expression r and
- a set of positive examples E^+,

find a regular expression r' such that

- $L(r) \subseteq L(r')$
- $E^+ \subseteq L(r')$

```
ISBN(978|979)?\d{10}
```

{ I978-2-1234-5680-3 }

Solution:

$r' = .^*$

Adding Words to Regexes: Problem statement

Problem statement:

Given

- a regular expression r,
- a set of positive examples E^+,
- a set of negative examples E^-,

find a regular expression r' such that

- $L(r) \subseteq L(r')$
- $E^+ \subseteq L(r')$
- $L(r') \cap E^-$ is small

```
ISBN(978|979)?\d{10}
```

{ I978-2-1234-5680-3 }

{ 0612345678 }

The YAGO knowledge base

Outline

Using YAGO for the humanities

Adding Words to Regexes

Introduction

Problem statement

What is new in our approach

Approximate regex matching

Finding the gaps

Add missing parts

Feedback function

Experiments

Summary

Answering Queries with Unix Shell

Conclusion

Adding Words to Regexes: Problem statement

Problem statement:

Given

- a regular expression r,
- a set of positive examples E^+,
- a set of negative examples E^-,

find a regular expression r' such that

- $L(r) \subseteq L(r')$
- $E^+ \subseteq L(r')$
- $L(r') \cap E^-$ is small

```
ISBN(978|979)?\d{10}
{ I978-2-1234-5680-3 }
{ 0612345678 }
```

Evaluation:

- Precision of $r' \geq$ or $\approx$ precision of r
- Recall of $r' \geq$ recall of r
  (w.r.t. the intended meaning of the regex)

The YAGO knowledge base
Outline
Using YAGO for the humanities
Adding Words to Regexes
Introduction
Problem statement
What is new in our approach
Approximate regex matching
Finding the gaps
Add missing parts
Feedback function
Experiments
Summary
Answering Queries with Unix Shell
Conclusion

Adding Words to Regexes: What is new in our approach

The YAGO knowledge base

Outline

Using YAGO for the humanities

Adding Words to Regexes

Introduction

Problem statement

What is new in our approach

Approximate regex matching

Finding the gaps

Add missing parts

Feedback function

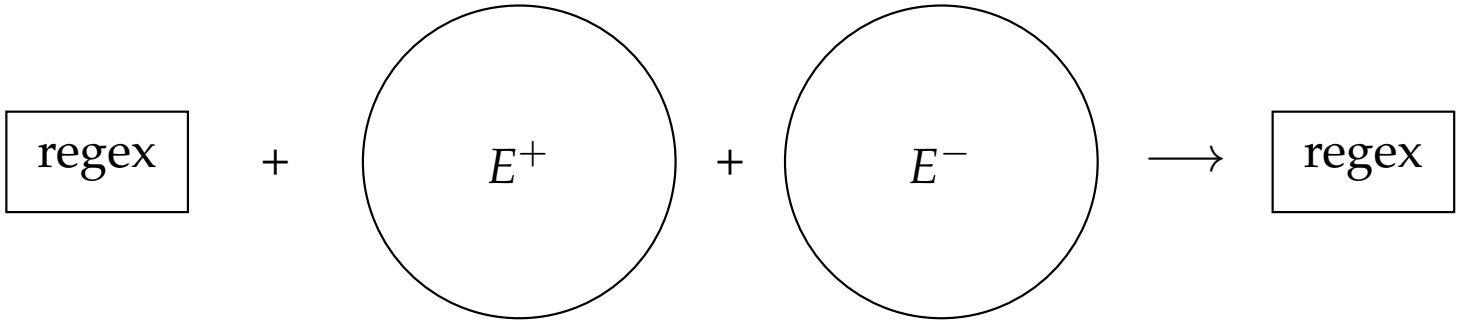
Experiments

Summary

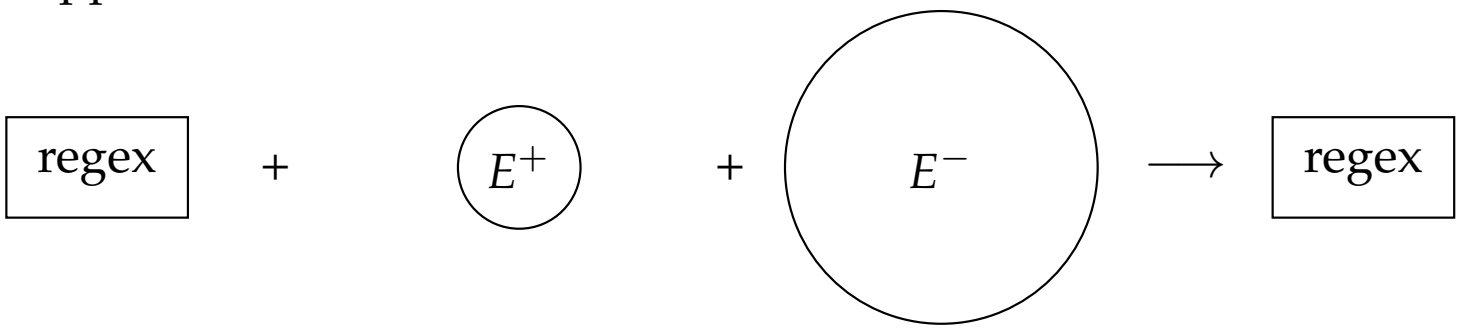
Answering Queries with Unix Shell

Conclusion

Previous approaches:



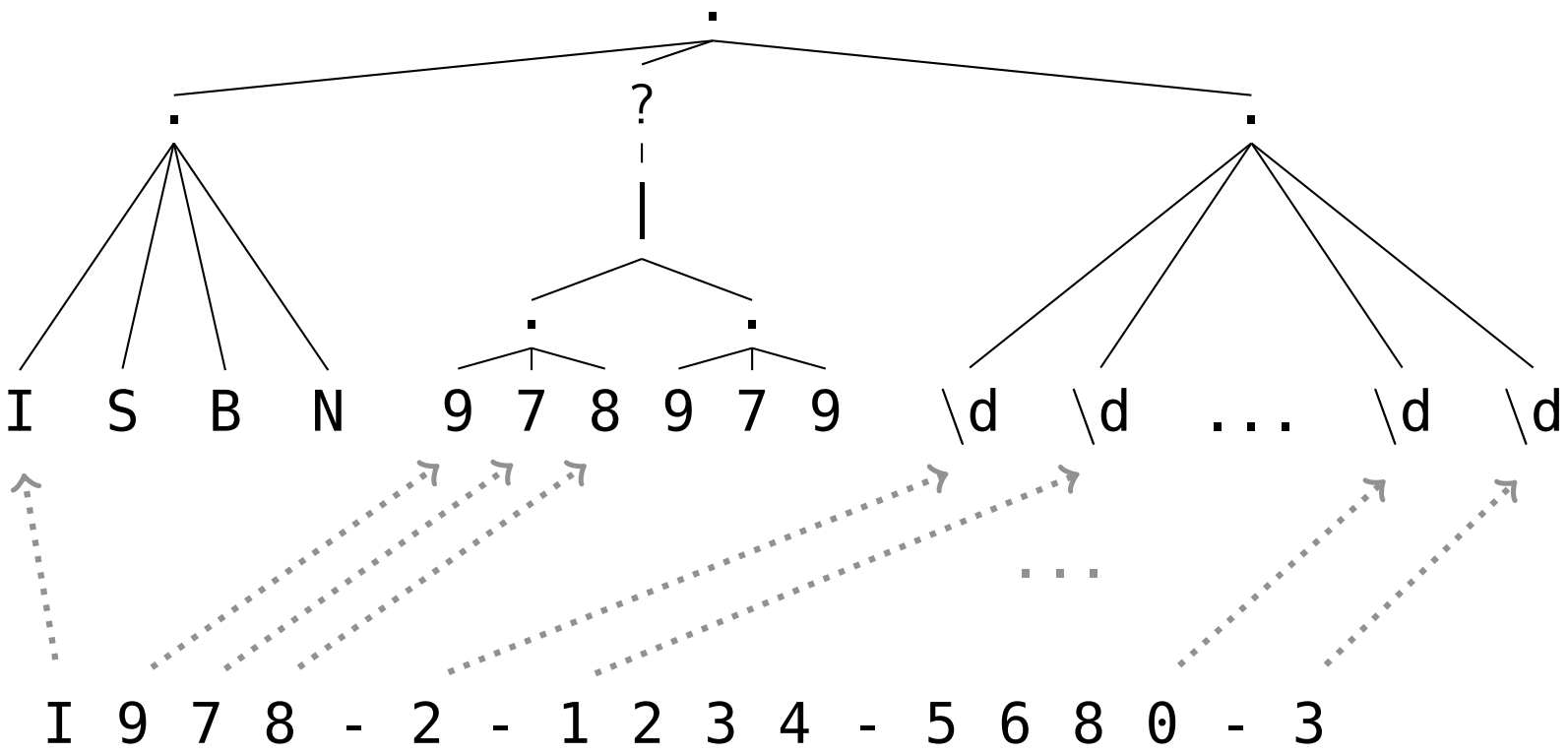
Our approach:



Rationale: creating a large set of positive examples is difficult

Adding Words to Regexes: Approximate regex matching

Step 1: match string and regex approximately [Myers et al. 1989]



Expanding the YAGO knowledge base

Rebele

The YAGO knowledge base

Outline

Using YAGO for the humanities

Adding Words to Regexes

Introduction

Problem statement

What is new in our approach

Approximate regex matching

Finding the gaps

Add missing parts

Feedback function

Experiments

Summary

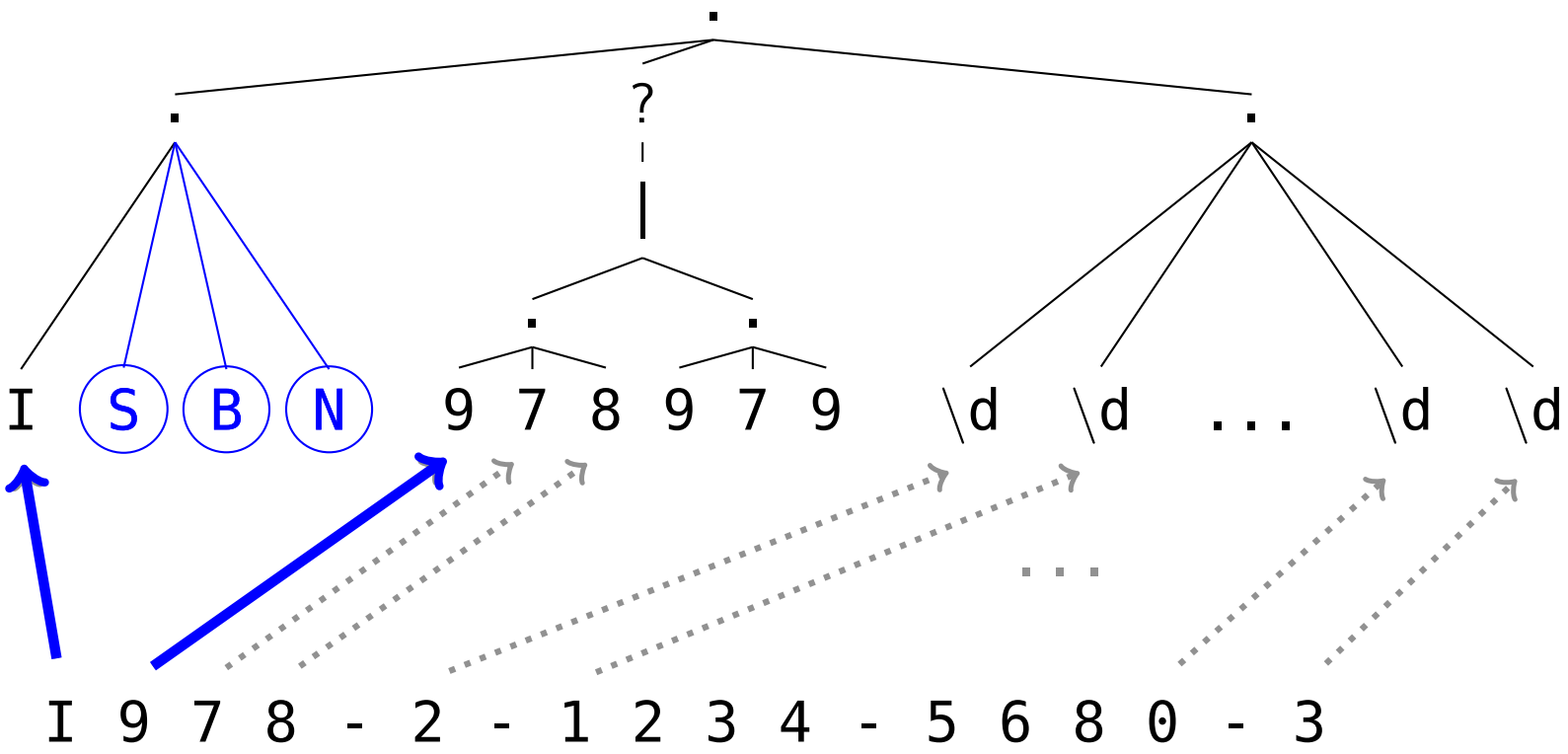
Answering Queries with Unix Shell

Conclusion

Adding Words to Regexes: Finding the gaps

Step 2: find the gaps

- Between regex leaves



The YAGO knowledge base

Outline

Using YAGO for the humanities

Adding Words to Regexes

Introduction

Problem statement

What is new in our approach

Approximate regex matching

Finding the gaps

Add missing parts

Feedback function

Experiments

Summary

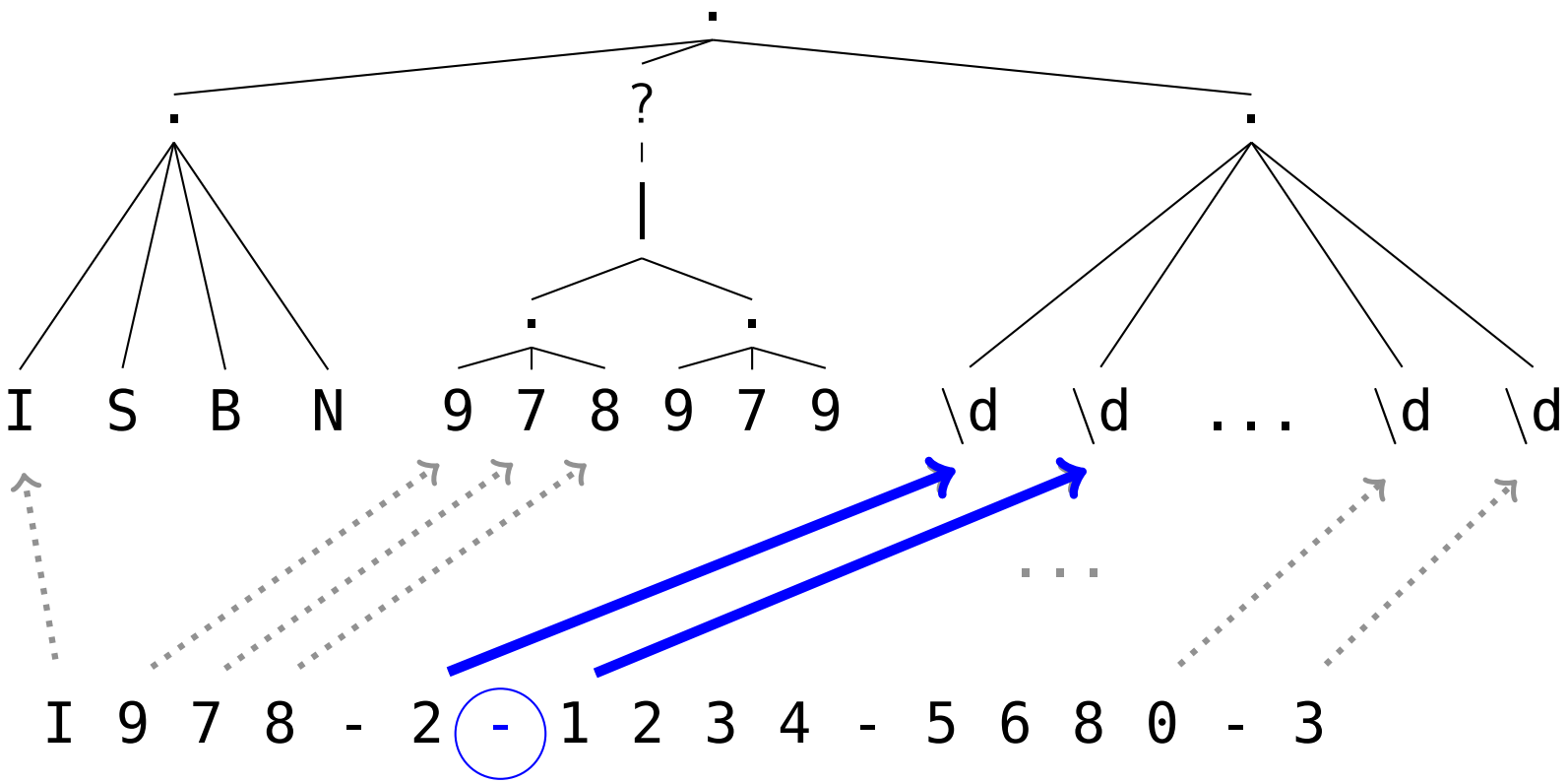
Answering Queries with Unix Shell

Conclusion

Adding Words to Regexes: Finding the gaps

Step 2: find the gaps

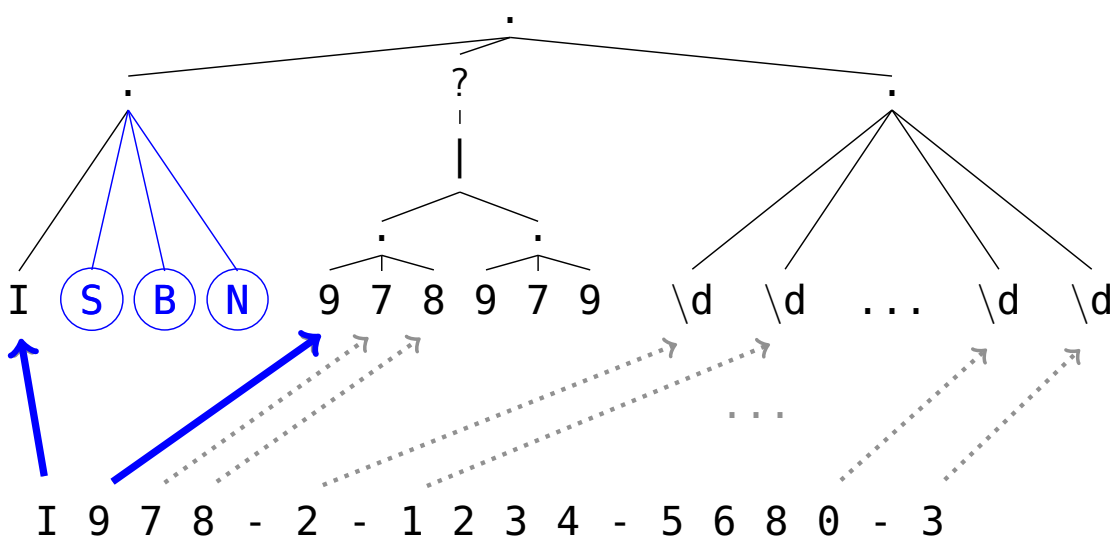
- Between regex leaves
- Between characters of the string

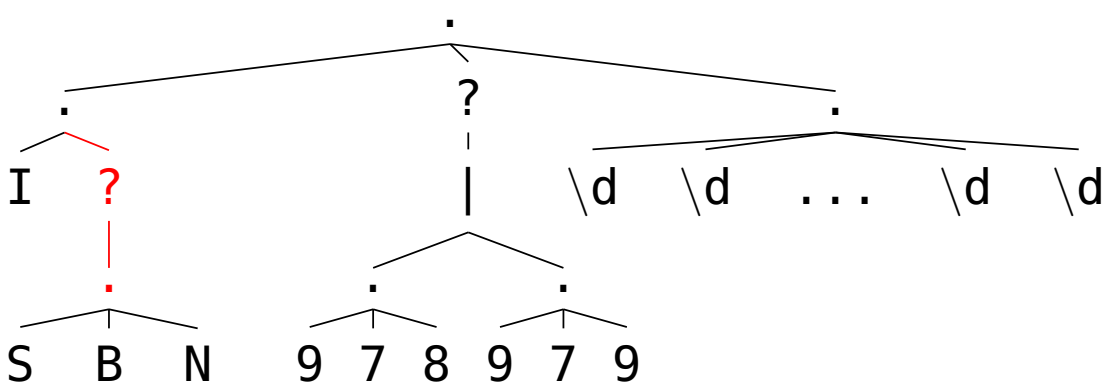


The YAGO knowledge base
Outline
Using YAGO for the humanities
Adding Words to Regexes
Introduction
Problem statement
What is new in our approach
Approximate regex matching
Finding the gaps
Add missing parts
Feedback function
Experiments
Summary
Answering Queries with Unix Shell
Conclusion

Adding Words to Regexes: Add missing parts

Step 3 (simple approach): adapt regex, so that it includes the missing parts

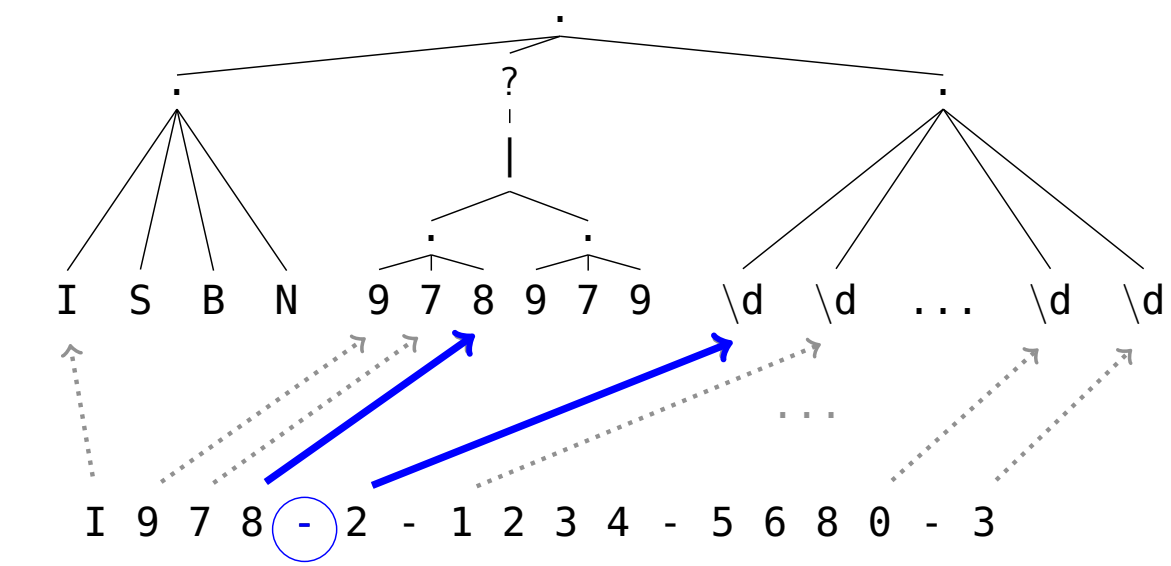


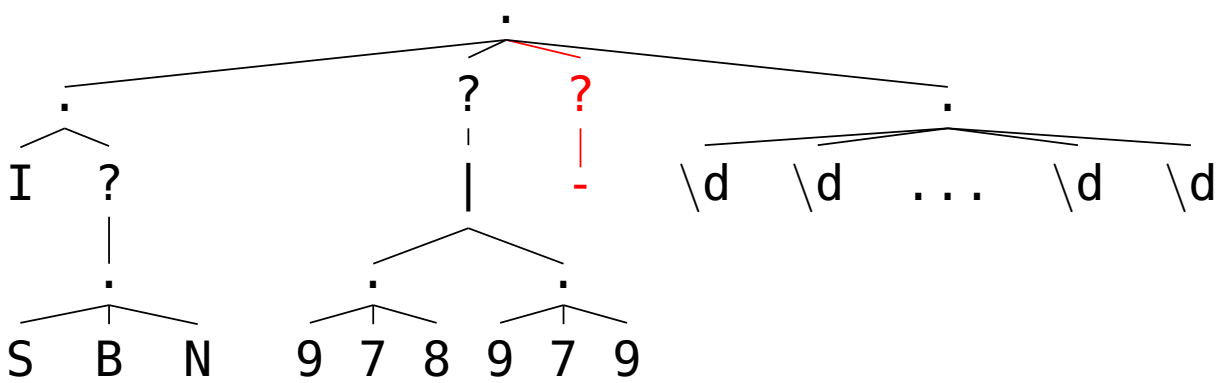


The YAGO knowledge base
Outline
Using YAGO for the humanities
Adding Words to Regexes
Introduction
Problem statement
What is new in our approach
Approximate regex matching
Finding the gaps
Add missing parts
Feedback function
Experiments
Summary
Answering Queries with Unix Shell
Conclusion

Adding Words to Regexes: Add missing parts

Step 3 (simple approach): adapt regex, so that it includes the missing parts

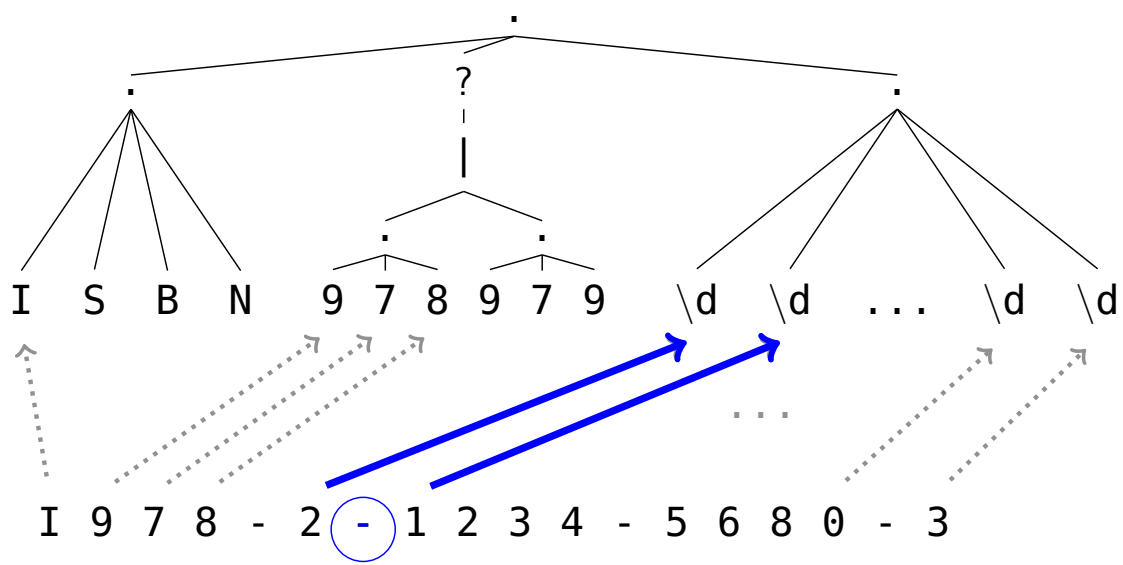


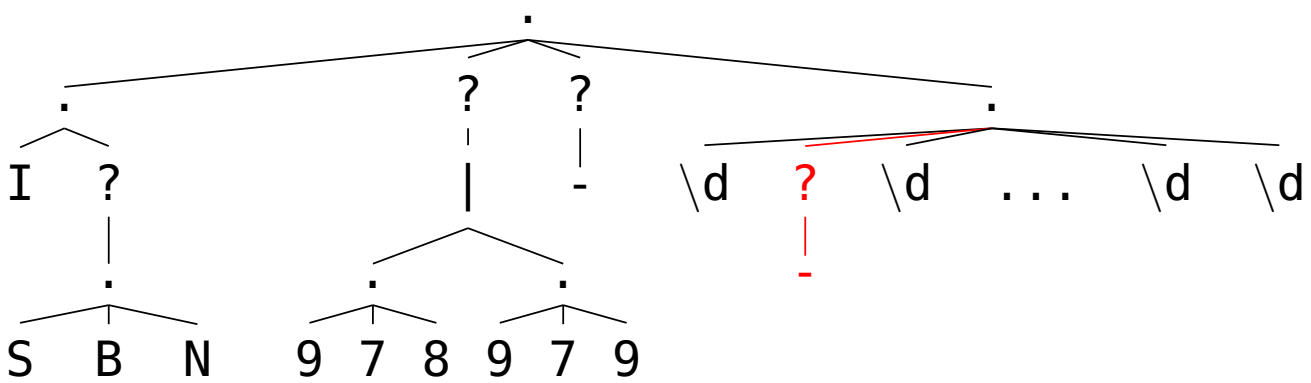


The YAGO knowledge base
Outline
Using YAGO for the humanities
Adding Words to Regexes
Introduction
Problem statement
What is new in our approach
Approximate regex matching
Finding the gaps
Add missing parts
Feedback function
Experiments
Summary
Answering Queries with Unix Shell
Conclusion

Adding Words to Regexes: Add missing parts

Step 3 (simple approach): adapt regex, so that it includes the missing parts

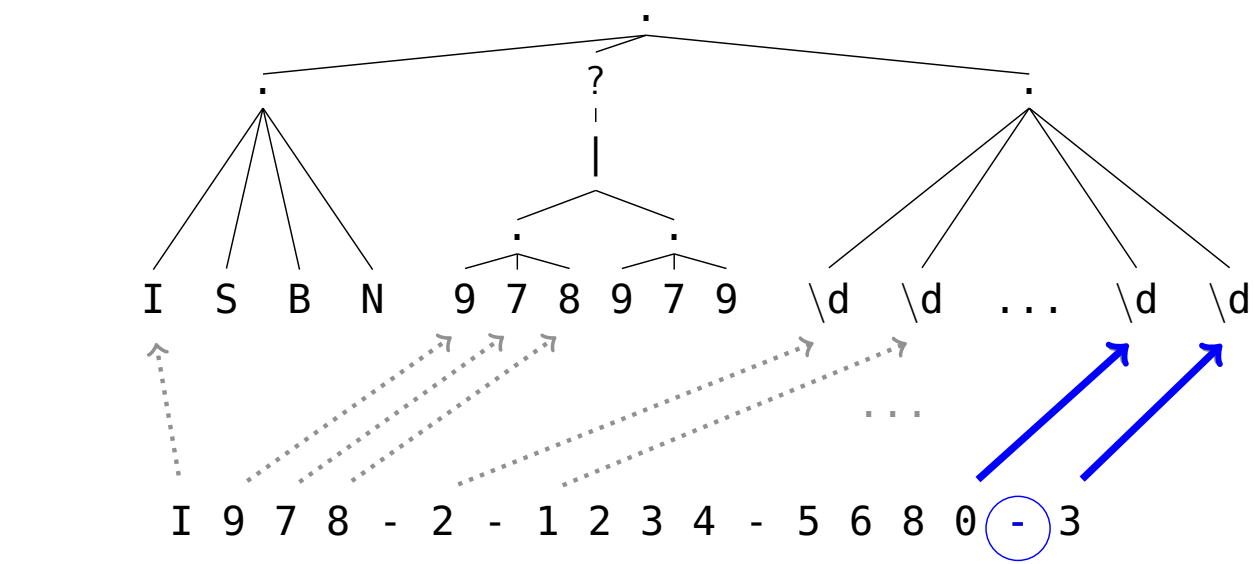


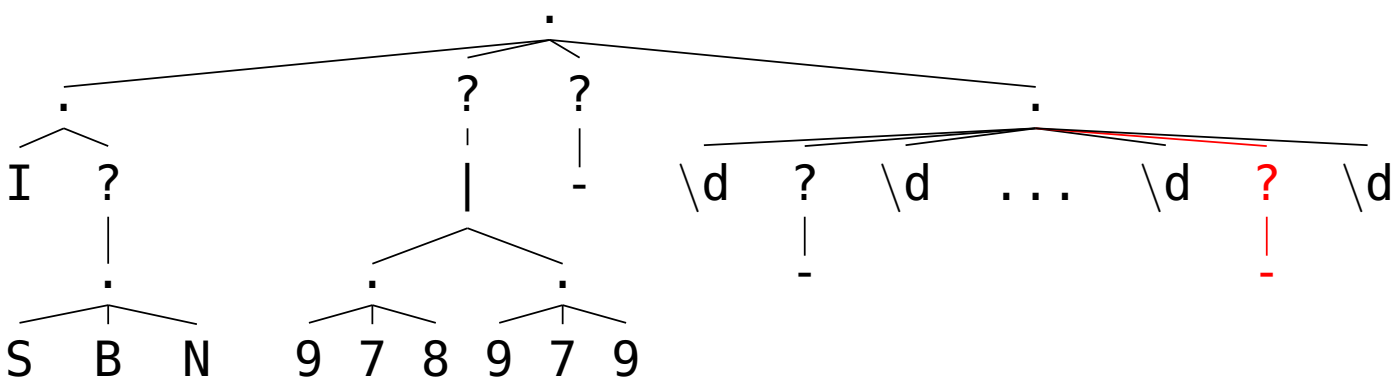


The YAGO knowledge base
Outline
Using YAGO for the humanities
Adding Words to Regexes
Introduction
Problem statement
What is new in our approach
Approximate regex matching
Finding the gaps
Add missing parts
Feedback function
Experiments
Summary
Answering Queries with Unix Shell
Conclusion

Adding Words to Regexes: Add missing parts

Step 3 (simple approach): adapt regex, so that it includes the missing parts



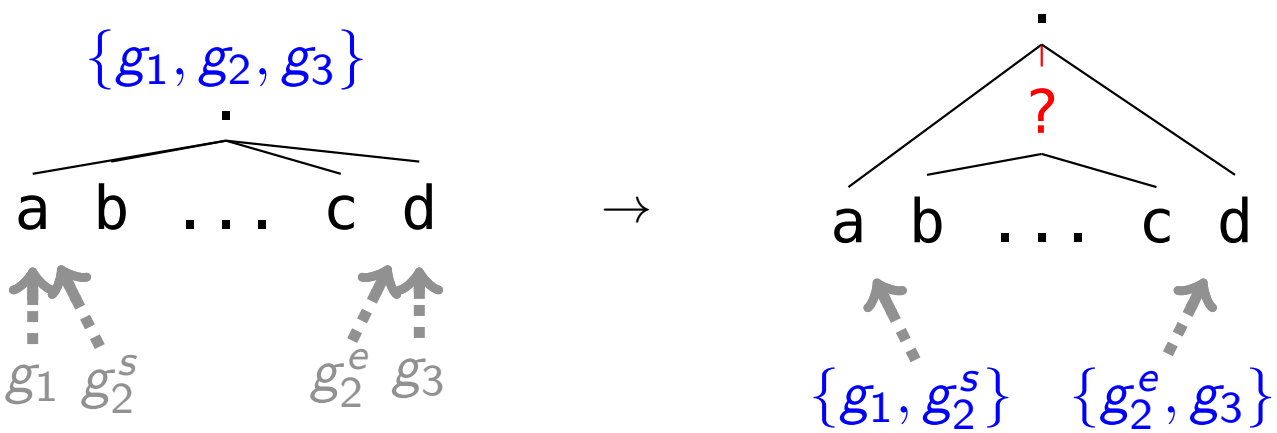


The YAGO knowledge base
Outline
Using YAGO for the humanities
Adding Words to Regexes
Introduction
Problem statement
What is new in our approach
Approximate regex matching
Finding the gaps
Add missing parts
Feedback function
Experiments
Summary
Answering Queries with Unix Shell
Conclusion

Adding Words to Regexes: Add missing parts

Step 3 (adaptive approach): adapt regex, so that it includes the missing parts

Exemplarily for a concatenation:

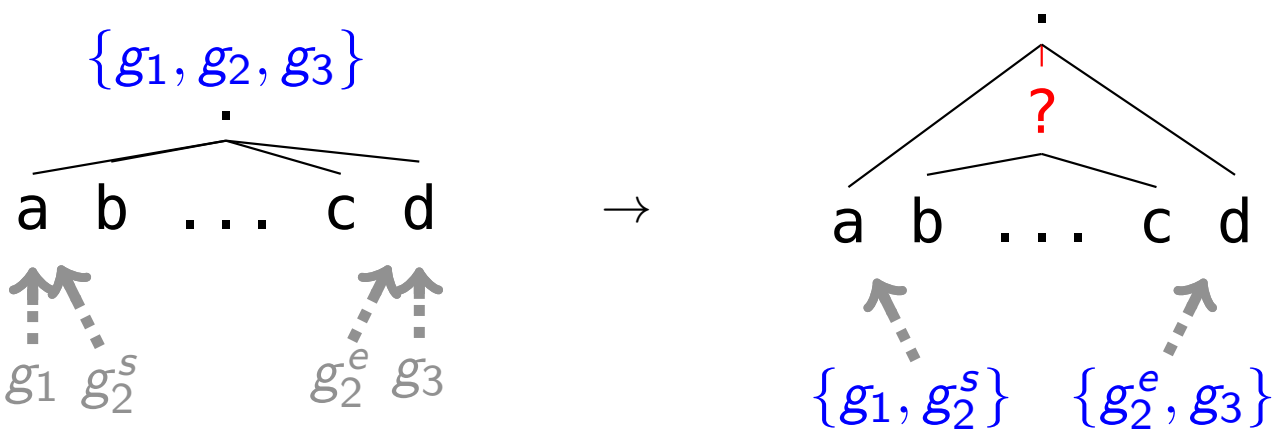


The YAGO knowledge base
Outline
Using YAGO for the humanities
Adding Words to Regexes
Introduction
Problem statement
What is new in our approach
Approximate regex matching
Finding the gaps
Add missing parts
Feedback function
Experiments
Summary
Answering Queries with Unix Shell
Conclusion

Adding Words to Regexes: Add missing parts

Step 3 (adaptive approach): adapt regex, so that it includes the missing parts

Exemplarily for a concatenation:



The YAGO knowledge base
Outline
Using YAGO for the humanities
Adding Words to Regexes
Introduction
Problem statement
What is new in our approach
Approximate regex matching
Finding the gaps
Add missing parts
Feedback function
Experiments
Summary
Answering Queries with Unix Shell
Conclusion

Adding Words to Regexes: Feedback function

Now we want to find URLs:

- We try regex $r =$ `http://[a-zA-Z\.]+`
- It does not find $s =$ `wikipedia.org`
- Repaired regex $r' =$ `(http://)?[a-zA-Z\.]+`

Problem:

- r' finds all words
- Precision drops

The YAGO knowledge base

Outline

Using YAGO for the humanities

Adding Words to Regexes

Introduction

Problem statement

What is new in our approach

Approximate regex matching

Finding the gaps

Add missing parts

Feedback function

Experiments

Summary

Answering Queries with Unix Shell

Conclusion

Adding Words to Regexes: Feedback function

Now we want to find URLs:

- We try regex $r =$ `http://[a-zA-Z\.]+`
- It does not find $s =$ `wikipedia.org`
- Repaired regex $r' =$ `(http://)?[a-zA-Z\.]+`

Problem:

- r' finds all words
- Precision drops

Solution: use feedback on set of negative examples E^-

- Determine the parts of the regex that we can make optional
- We use the number of false positives, i.e.,

$$f(r') = |E^- \cap L(r')| \leq \alpha |E^- \cap L(r)|$$

- If $f(r') =$ false, add the word as disjunction instead: `http://[a-zA-Z\.]+|wikipedia.org`

The YAGO knowledge base
Outline
Using YAGO for the humanities
Adding Words to Regexes
Introduction
Problem statement
What is new in our approach
Approximate regex matching
Finding the gaps
Add missing parts
Feedback function
Experiments
Summary
Answering Queries with Unix Shell
Conclusion

Adding Words to Regexes: Experiments

Input data:

- Datasets:
  ReLIE [Li et al., 2008],
  Enron [Babbar et al., 2010], and
  Wikipedia infobox attributes
- In total 8 tasks (e.g., phone numbers, software names, dates)
- In total 52 regexes

Expanding the YAGO knowledge base
Rebele
The YAGO knowledge base
Outline
Using YAGO for the humanities
Adding Words to Regexes
Introduction
Problem statement
What is new in our approach
Approximate regex matching
Finding the gaps
Add missing parts
Feedback function
Experiments
Summary
Answering Queries with Unix Shell
Conclusion

Adding Words to Regexes: Experiments

Input data:

- Datasets:
  ReLIE [Li et al., 2008],
  Enron [Babbar et al., 2010], and
  Wikipedia infobox attributes
- In total 8 tasks (e.g., phone numbers, software names, dates)
- In total 52 regexes

Approaches:

- Dis: $r|s_1|\cdots|s_n$
- Star: `.*`
- B&S: [Babbar et al., 2010] (reimplementation)
- Simple
- Adaptive

		baseline				adaptive		
measure	original	dis	star	B&S	simple	$\alpha = 1.0$	$\alpha = 1.1$	$\alpha = 1.20$
F1	55	55	21	40	56	**60**	**60**	**60**
recall	66	67	62	35	69	75	76	**77**
precision	64	64	14	**71**	64	63	63	63
length	56	270	2	3929	250	**76**	80	81

Table: Averaged measures for the different systems. Length is # of characters of the regex.

The YAGO knowledge base
Outline
Using YAGO for the humanities
Adding Words to Regexes
Introduction
Problem statement
What is new in our approach
Approximate regex matching
Finding the gaps
Add missing parts
Feedback function
Experiments
Summary
Answering Queries with Unix Shell
Conclusion

Adding Words to Regexes: Summary

Summary:

- Algorithm for adding missing words to regexes
- Increases recall, while keeping precision stable
- Source code available at `https://github.com/thomasrebele/regex-repair`

Future work:

- Decrease dependency on E^-
- Add a generalization step as postprocessing

Publications: ISWC 2017 (demo), PAKDD 2018 (full paper)


Thomas Rebele


Katerina Tzompanaki


Fabian Suchanek

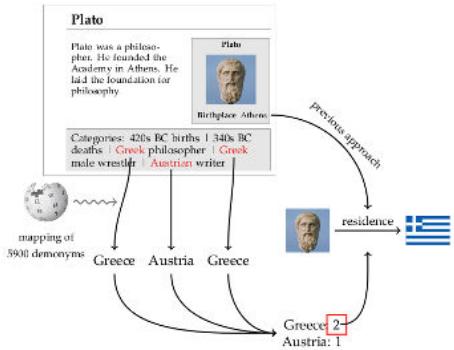
Now that we have all this data, how can we process it efficiently?

The YAGO knowledge base
Outline
Using YAGO for the humanities
Adding Words to Regexes
Introduction
Problem statement
What is new in our approach
Approximate regex matching
Finding the gaps
Add missing parts
Feedback function
Experiments
Summary
Answering Queries with Unix Shell
Conclusion

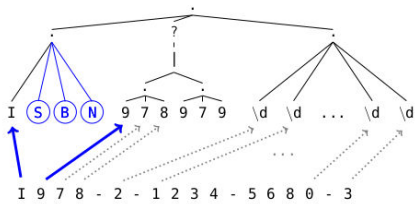
Adding Words to Regexes: Summary

Contributions:

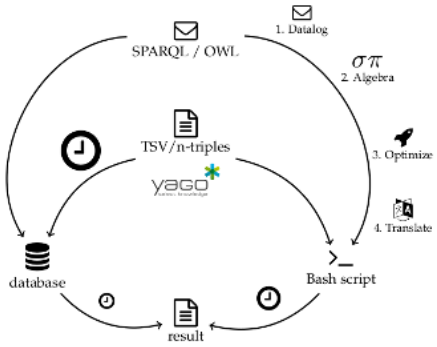
Extracting more information about residences, gender, birth and death dates



Repairing regular expressions by adding missing words

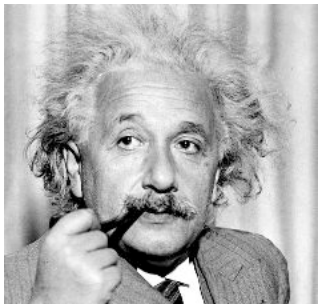


Preprocessing tabular data by transforming queries to Bash scripts

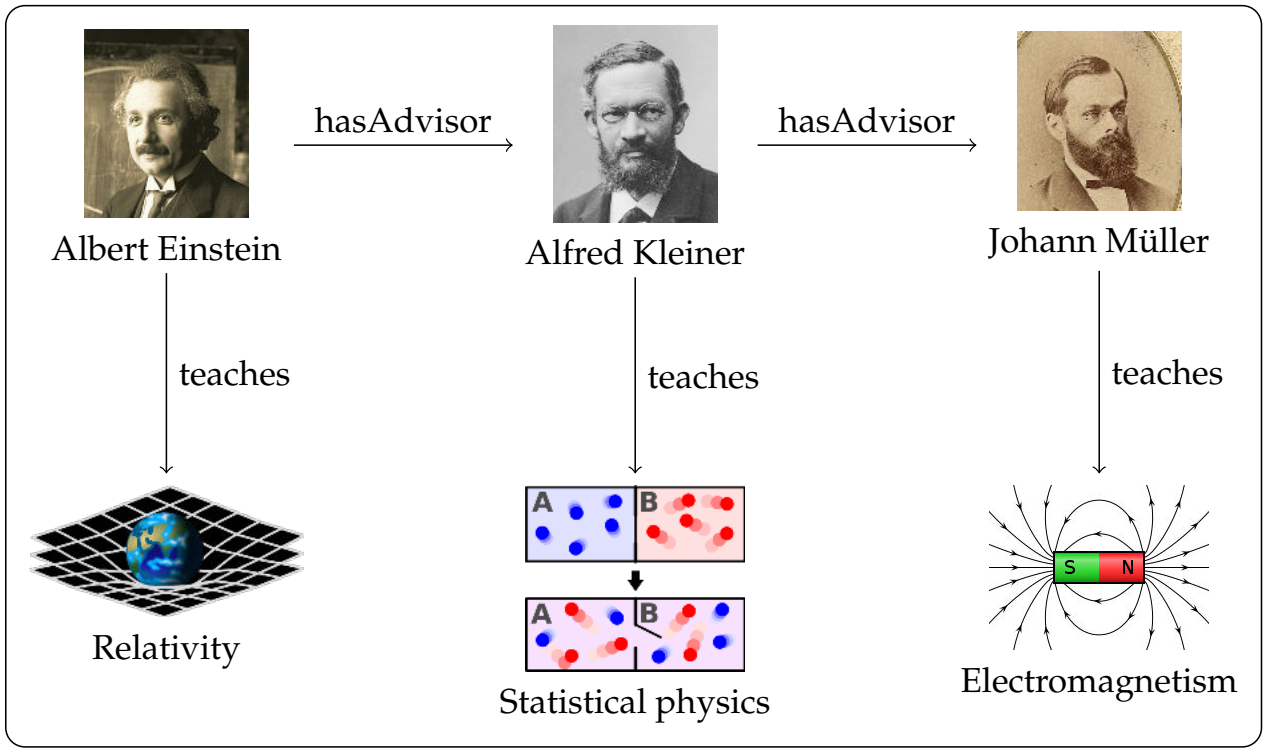


The YAGO knowledge base
Outline
Using YAGO for the humanities
Adding Words to Regexes
Introduction
Problem statement
What is new in our approach
Approximate regex matching
Finding the gaps
Add missing parts
Feedback function
Experiments
Summary
Answering Queries with Unix Shell
Conclusion

Answering Queries with Unix Shell: Motivation



How can I find
all my academic ancestors?



The YAGO knowledge base

Outline

Using YAGO for the humanities

Adding Words to Regexes

Answering Queries with Unix Shell

Motivation

Idea

Approach

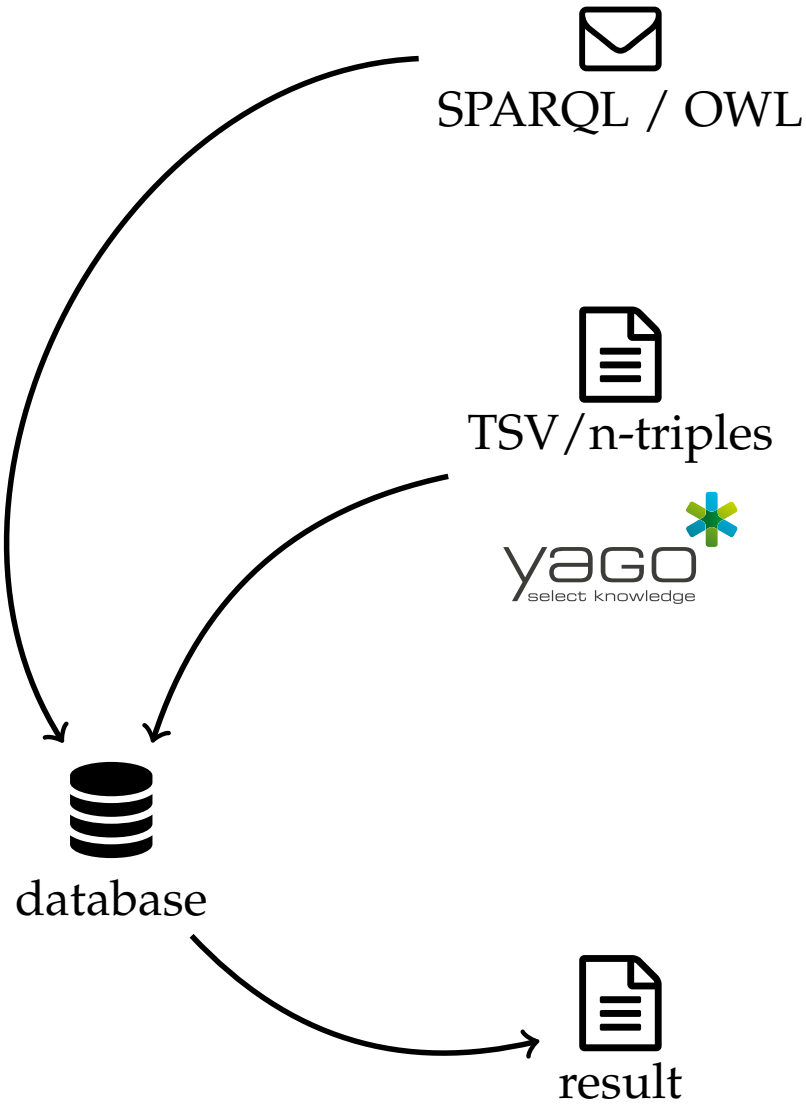
Optimization

Experiments

Summary

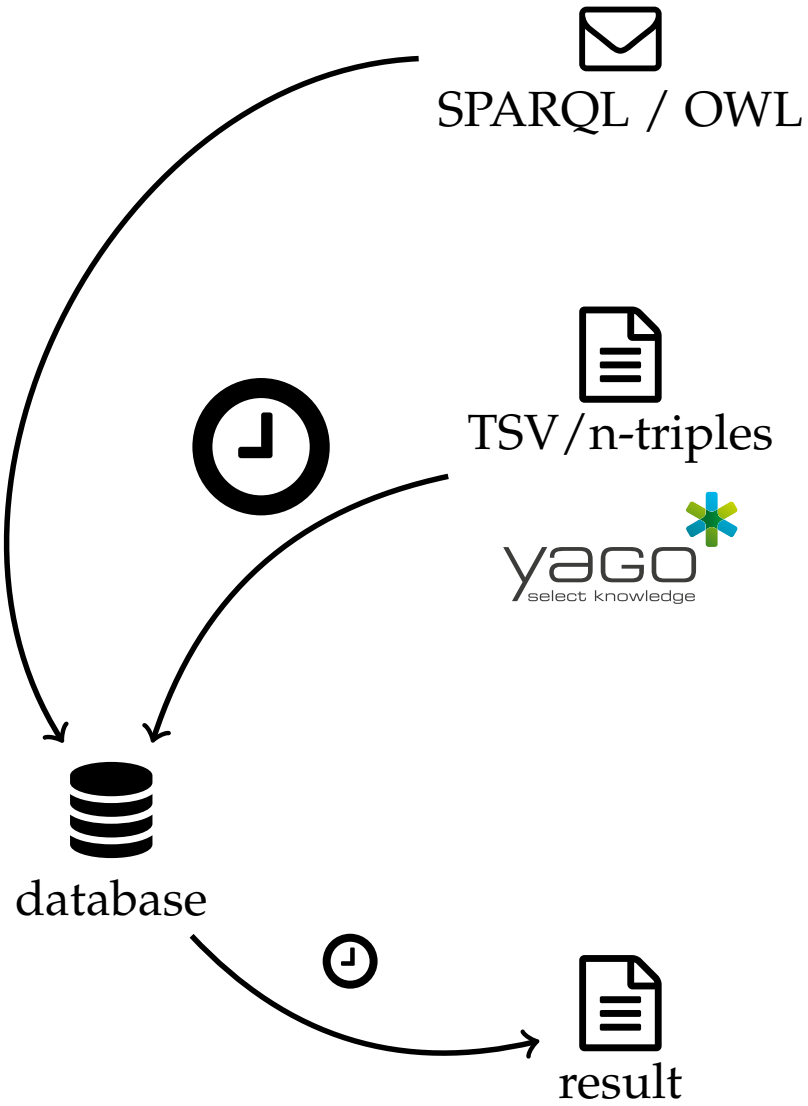
Conclusion

Answering Queries with Unix Shell: Idea



The YAGO knowledge base
Outline
Using YAGO for the humanities
Adding Words to Regexes
Answering Queries with Unix Shell
Motivation
Idea
Approach
Optimization
Experiments
Summary
Conclusion

Answering Queries with Unix Shell: Idea



The YAGO knowledge base

Outline

Using YAGO for the humanities

Adding Words to Regexes

Answering Queries with Unix Shell

Motivation

Idea

Approach

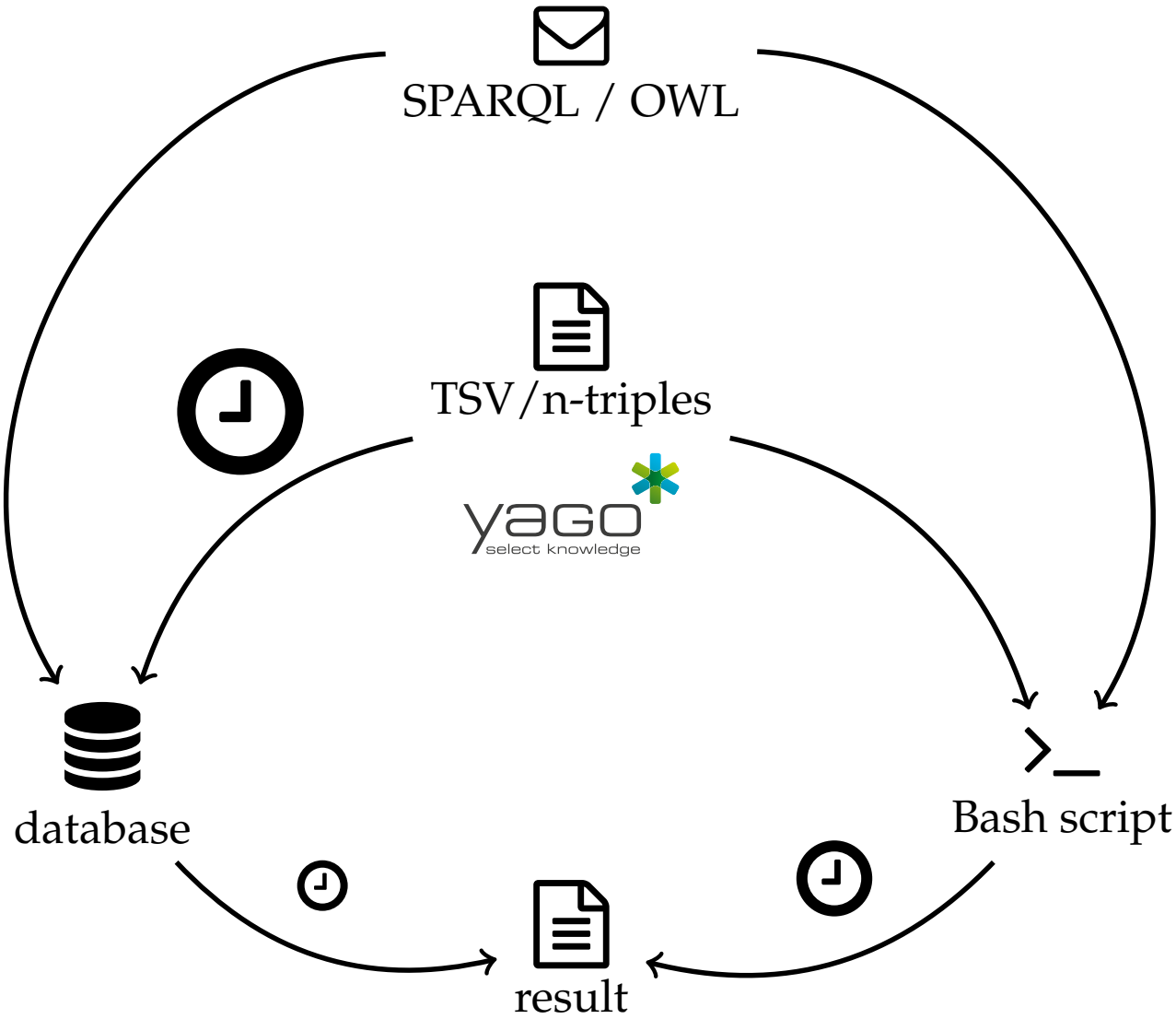
Optimization

Experiments

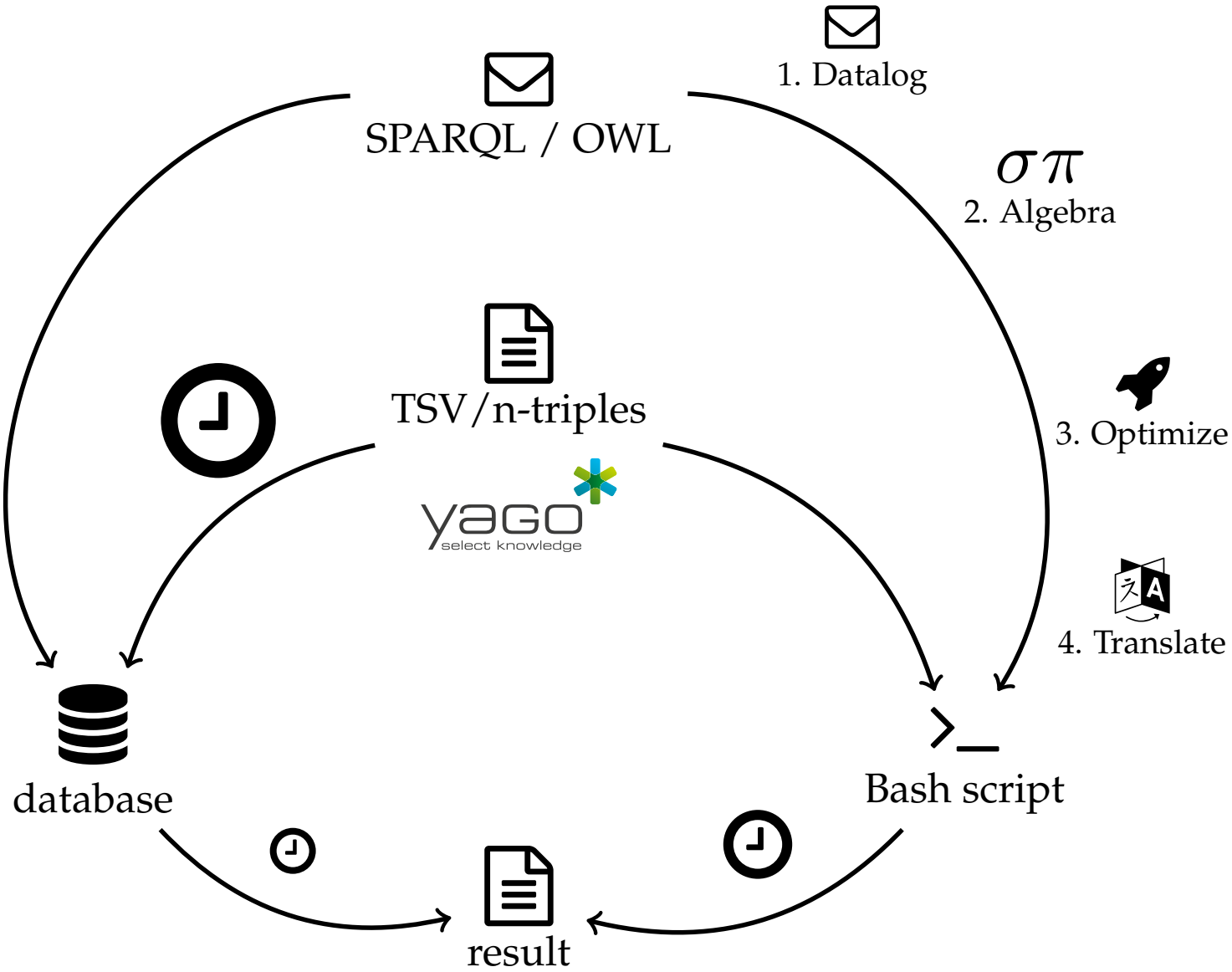
Summary

Conclusion

Answering Queries with Unix Shell: Idea



Answering Queries with Unix Shell: Idea



Expanding the YAGO knowledge base

Rebele

The YAGO knowledge base

Outline

Using YAGO for the humanities

Adding Words to Regexes

Answering Queries with Unix Shell

Motivation

Idea

Approach

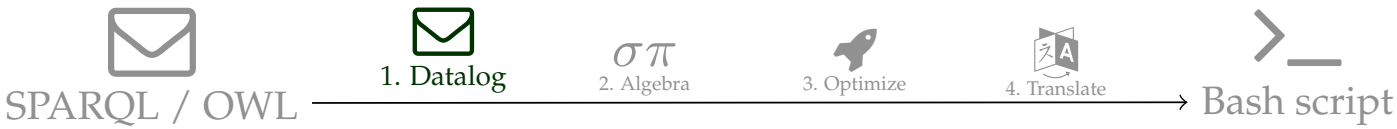
Optimization

Experiments

Summary

Conclusion

Answering Queries with Unix Shell: Approach



Query "Who are Einstein's academic ancestors?" in SPARQL:

```
SELECT ?Y WHERE {
    <Einstein> <hasAdvisor>+ ?Y
}
```

Translating the query to Datalog (simplified):

```
facts(X, Y, Z) :~ read_ntriples input

adv(X, Y) :- facts(X, "hasAdvisor", Y).

result(Y) :- adv("Einstein", Y).
result(Y) :- result(X), adv(X, Y).
```

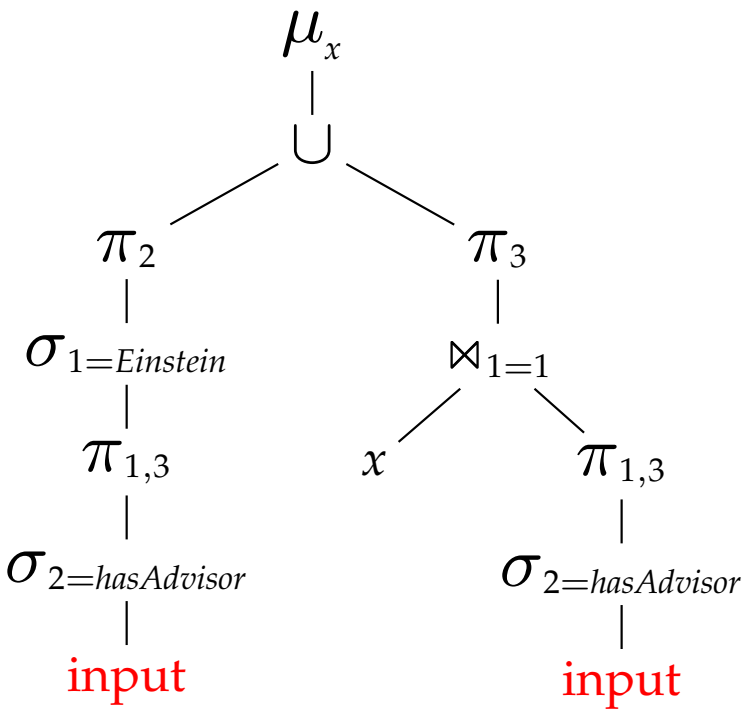

Expanding the YAGO knowledge base
Rebele
The YAGO knowledge base
Outline
Using YAGO for the humanities
Adding Words to Regexes
Answering Queries with Unix Shell
Motivation
Idea
Approach
Optimization
Experiments
Summary
Conclusion

Answering Queries with Unix Shell: Approach


```
facts(X, Y, Z) :~ read_ntriples input

adv(X, Y) :- facts(X, "hasAdvisor", Y).

result(Y) :- adv("Einstein", Y).
result(Y) :- result(X), adv(X, Y).
```



Expanding the YAGO knowledge base

Rebele

The YAGO knowledge base

Outline

Using YAGO for the humanities

Adding Words to Regexes

Answering Queries with Unix Shell

Motivation

Idea

Approach

Optimization

Experiments

Summary

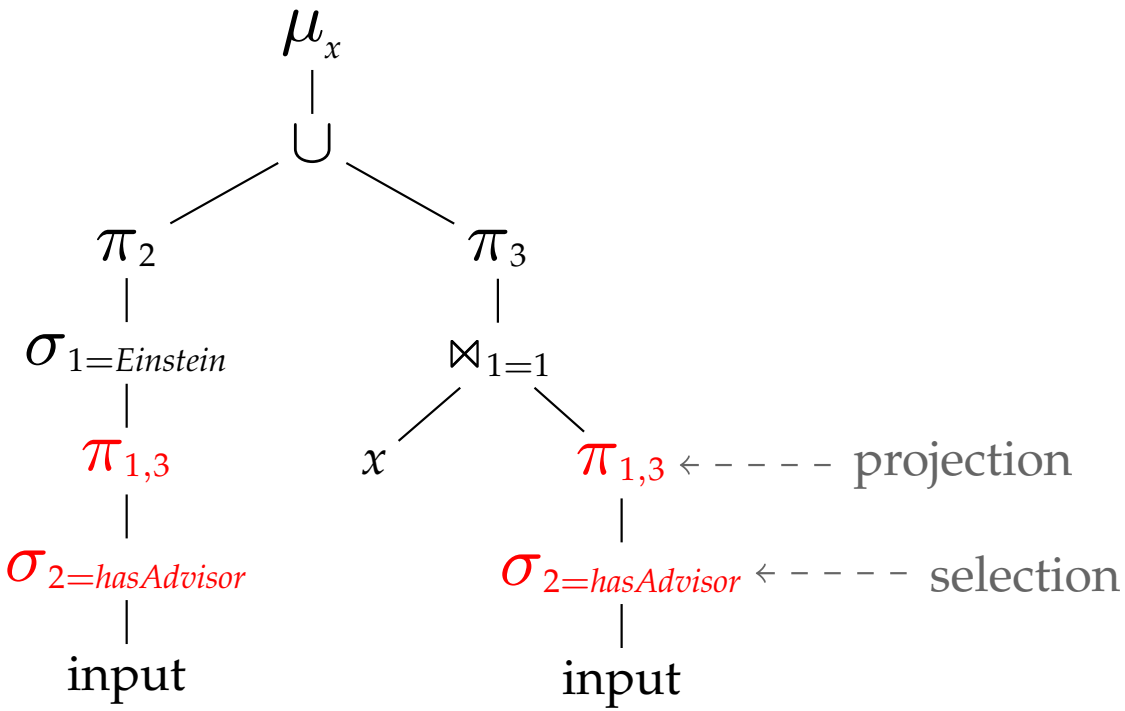
Conclusion

Answering Queries with Unix Shell: Approach


```
facts(X, Y, Z) :~ read_ntriples input

adv(X, Y) :- facts(X, "hasAdvisor", Y).

result(Y) :- adv("Einstein", Y).
result(Y) :- result(X), adv(X, Y).
```



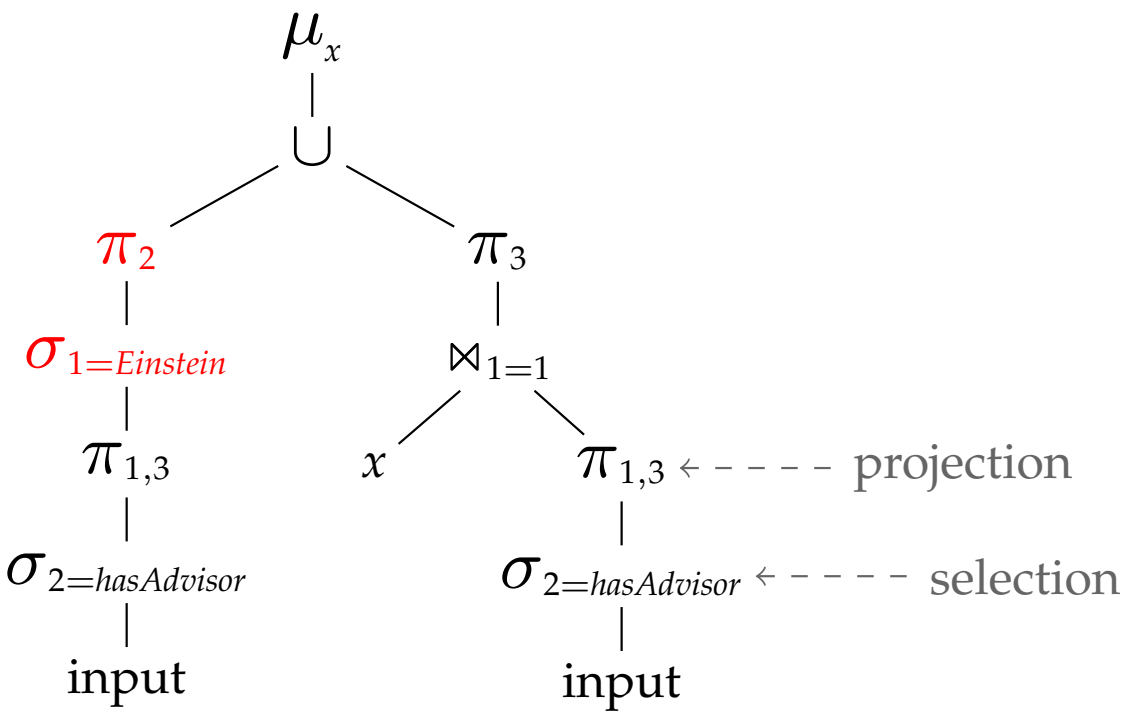
Expanding the YAGO knowledge base
Rebele
The YAGO knowledge base
Outline
Using YAGO for the humanities
Adding Words to Regexes
Answering Queries with Unix Shell
Motivation
Idea
Approach
Optimization
Experiments
Summary
Conclusion

Answering Queries with Unix Shell: Approach


```
facts(X, Y, Z) :~ read_ntriples input

adv(X, Y) :- facts(X, "hasAdvisor", Y).

result(Y) :- adv("Einstein", Y).
result(Y) :- result(X), adv(X, Y).
```



Expanding the YAGO knowledge base

Rebele

The YAGO knowledge base

Outline

Using YAGO for the humanities

Adding Words to Regexes

Answering Queries with Unix Shell

Motivation

Idea

Approach

Optimization

Experiments

Summary

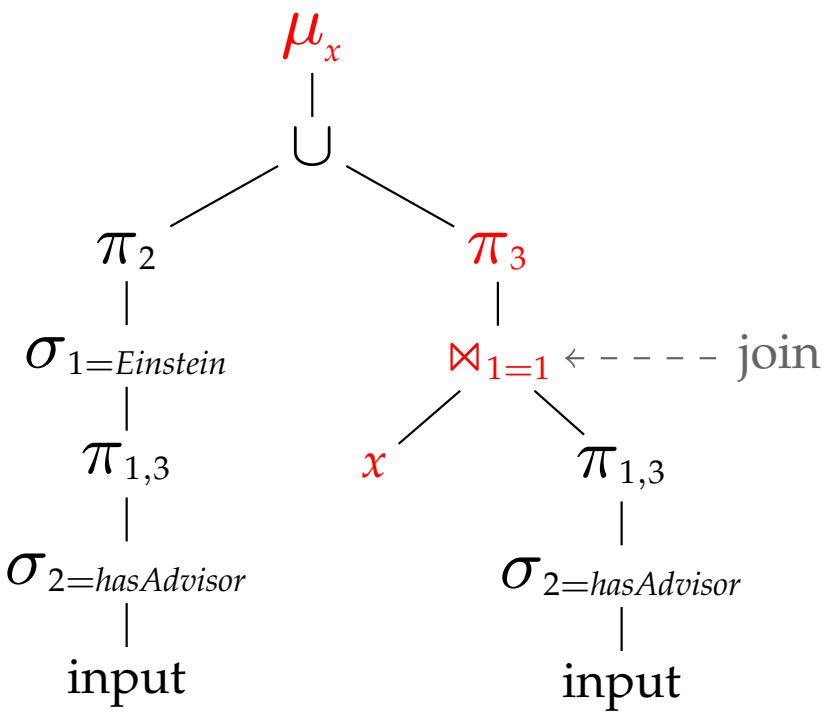
Conclusion

Answering Queries with Unix Shell: Approach


```
facts(X, Y, Z) :~ read_ntriples input

adv(X, Y) :- facts(X, "hasAdvisor", Y).

result(Y) :- adv("Einstein", Y).
result(Y) :- result(X), adv(X, Y).
```



Expanding the YAGO knowledge base

Rebele

The YAGO knowledge base

Outline

Using YAGO for the humanities

Adding Words to Regexes

Answering Queries with Unix Shell

Motivation

Idea

Approach

Optimization

Experiments

Summary

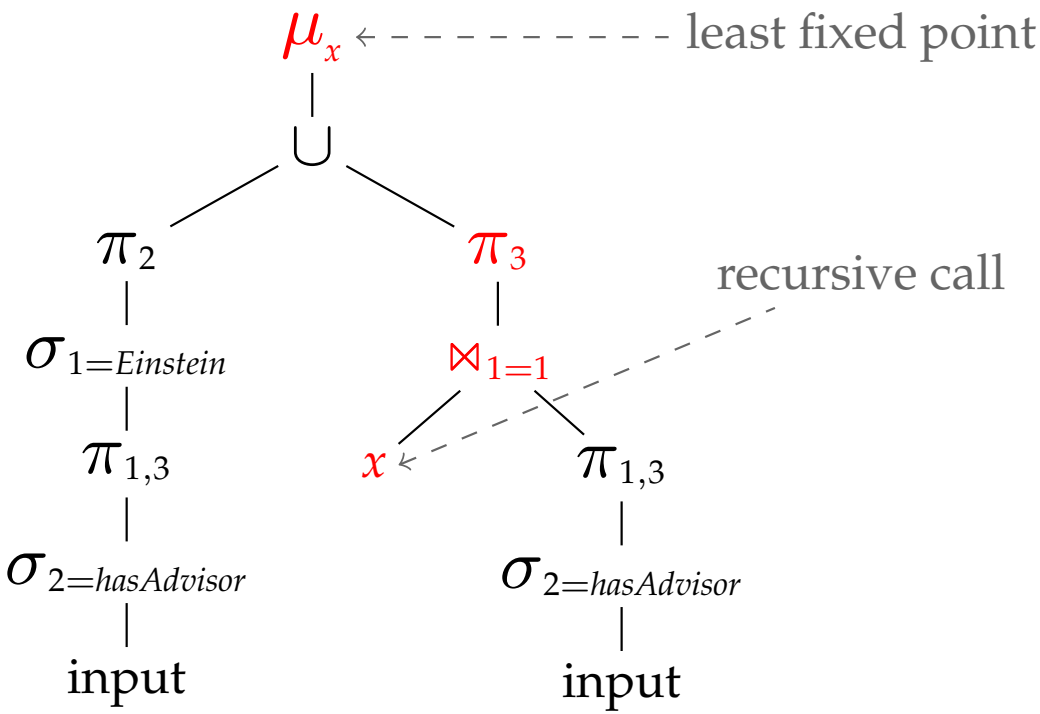
Conclusion

Answering Queries with Unix Shell: Approach


```
facts(X, Y, Z) :~ read_ntriples input

adv(X, Y) :- facts(X, "hasAdvisor", Y).

result(Y) :- adv("Einstein", Y).
result(Y) :- result(X), adv(X, Y).
```



Expanding the YAGO knowledge base

Rebele

The YAGO knowledge base

Outline

Using YAGO for the humanities

Adding Words to Regexes

Answering Queries with Unix Shell

Motivation

Idea

Approach

Optimization

Experiments

Summary

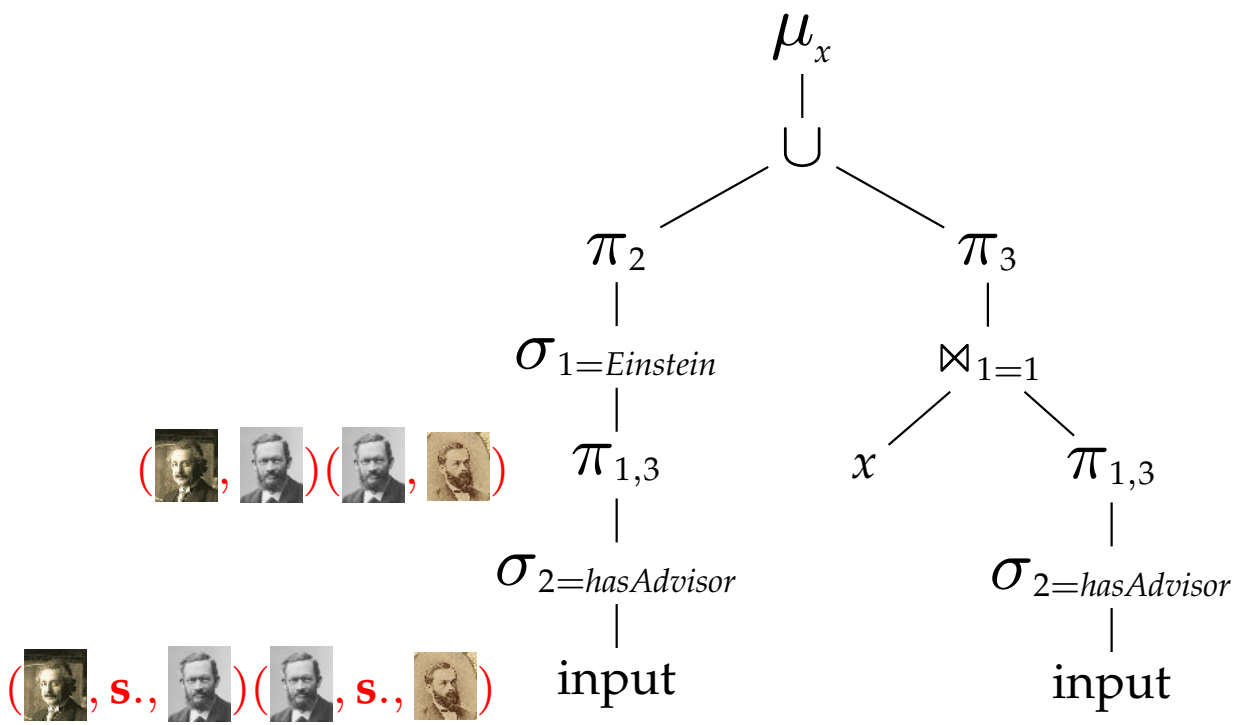
Conclusion

Answering Queries with Unix Shell: Approach


```
facts(X, Y, Z) :~ read_ntriples input

adv(X, Y) :- facts(X, "hasAdvisor", Y).

result(Y) :- adv("Einstein", Y).
result(Y) :- result(X), adv(X, Y).
```



Expanding the YAGO knowledge base

Rebele

The YAGO knowledge base

Outline

Using YAGO for the humanities

Adding Words to Regexes

Answering Queries with Unix Shell

Motivation

Idea

Approach

Optimization

Experiments

Summary

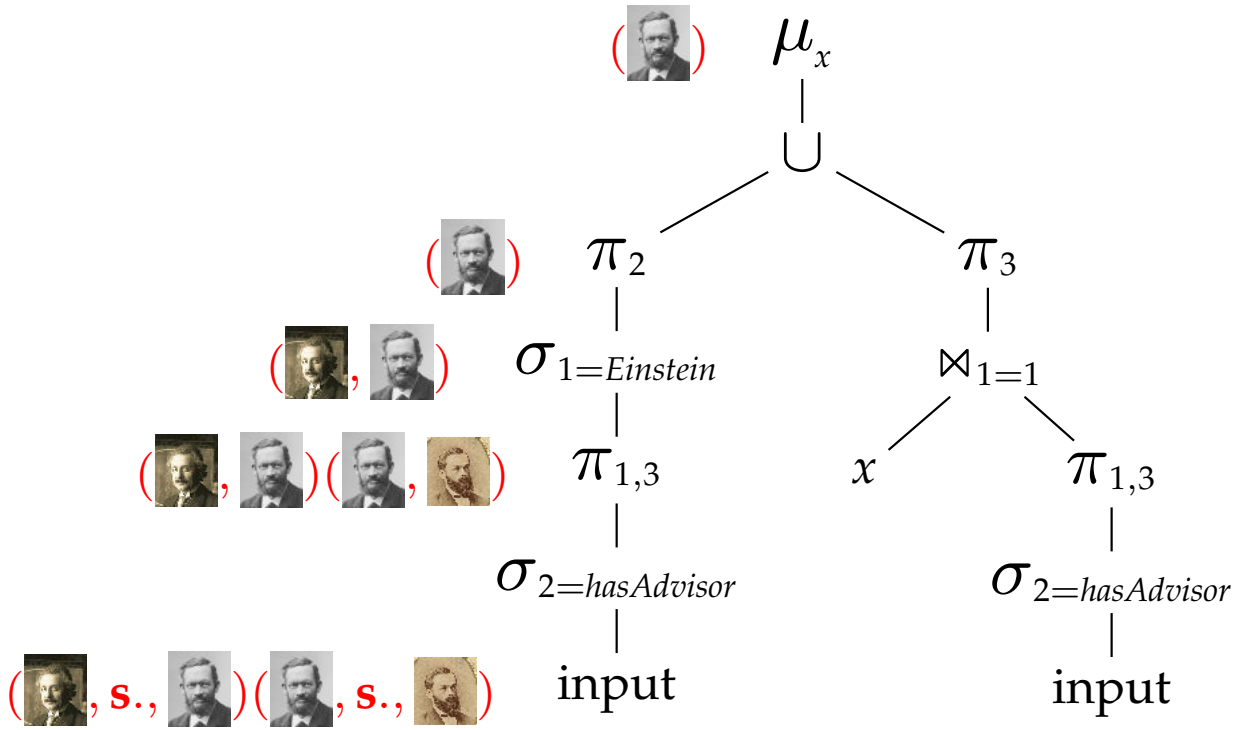
Conclusion

Answering Queries with Unix Shell: Approach


```
facts(X, Y, Z) :~ read_ntriples input

adv(X, Y) :- facts(X, "hasAdvisor", Y).

result(Y) :- adv("Einstein", Y).
result(Y) :- result(X), adv(X, Y).
```



Expanding the YAGO knowledge base

Rebele

The YAGO knowledge base

Outline

Using YAGO for the humanities

Adding Words to Regexes

Answering Queries with Unix Shell

Motivation

Idea

Approach

Optimization

Experiments

Summary

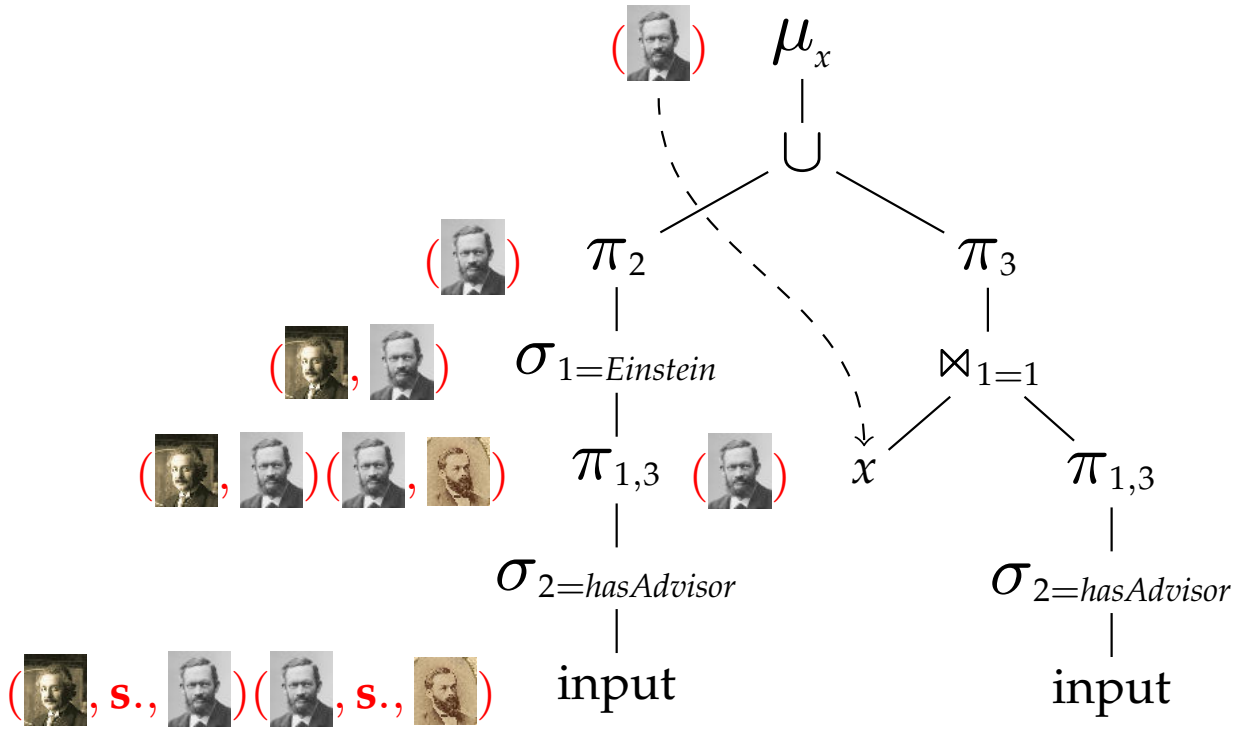
Conclusion

Answering Queries with Unix Shell: Approach


```
facts(X, Y, Z) :~ read_ntriples input

adv(X, Y) :- facts(X, "hasAdvisor", Y).

result(Y) :- adv("Einstein", Y).
result(Y) :- result(X), adv(X, Y).
```



Expanding the YAGO knowledge base

Rebele

The YAGO knowledge base

Outline

Using YAGO for the humanities

Adding Words to Regexes

Answering Queries with Unix Shell

Motivation

Idea

Approach

Optimization

Experiments

Summary

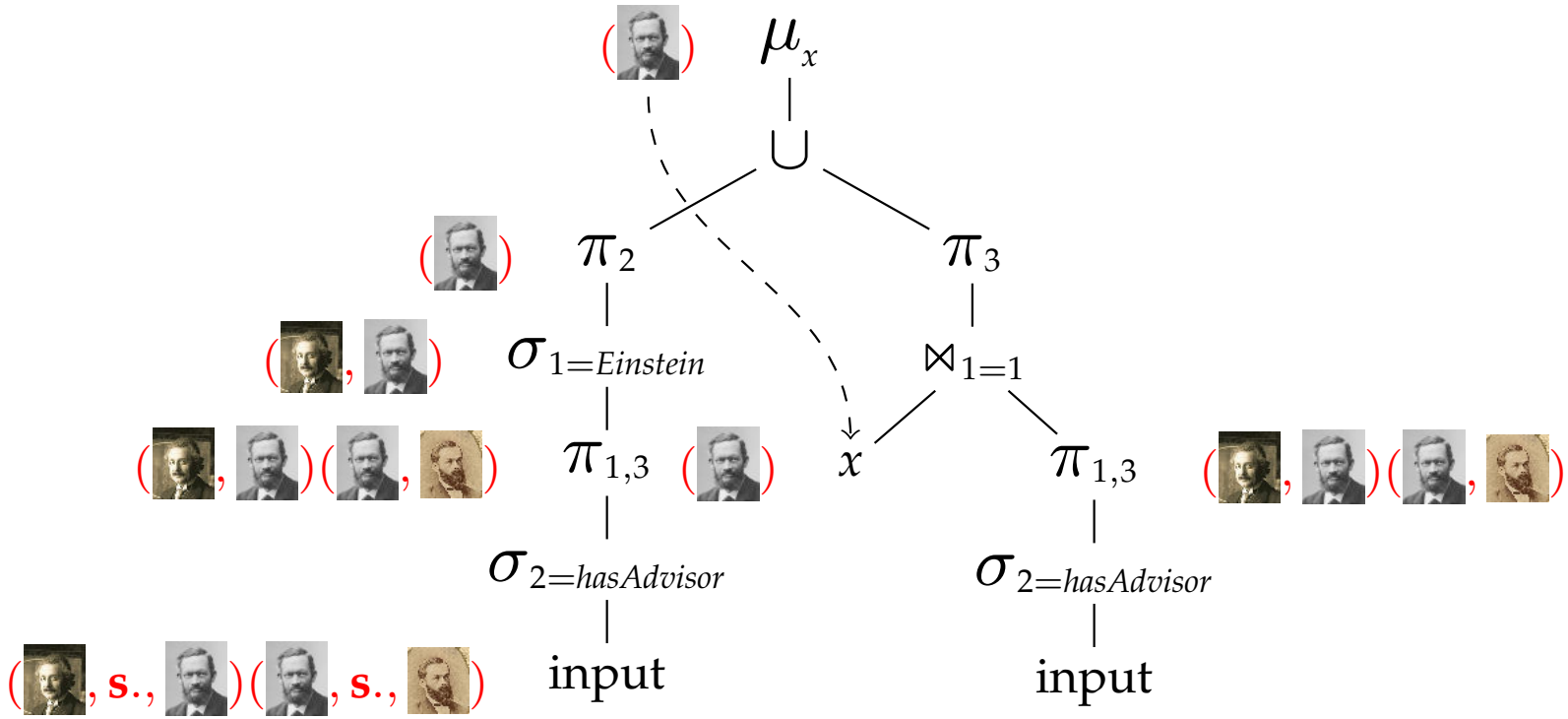
Conclusion

Answering Queries with Unix Shell: Approach


```
facts(X, Y, Z) :~ read_ntriples input

adv(X, Y) :- facts(X, "hasAdvisor", Y).

result(Y) :- adv("Einstein", Y).
result(Y) :- result(X), adv(X, Y).
```



Expanding the YAGO knowledge base

Rebele

The YAGO knowledge base

Outline

Using YAGO for the humanities

Adding Words to Regexes

Answering Queries with Unix Shell

Motivation

Idea

Approach

Optimization

Experiments

Summary

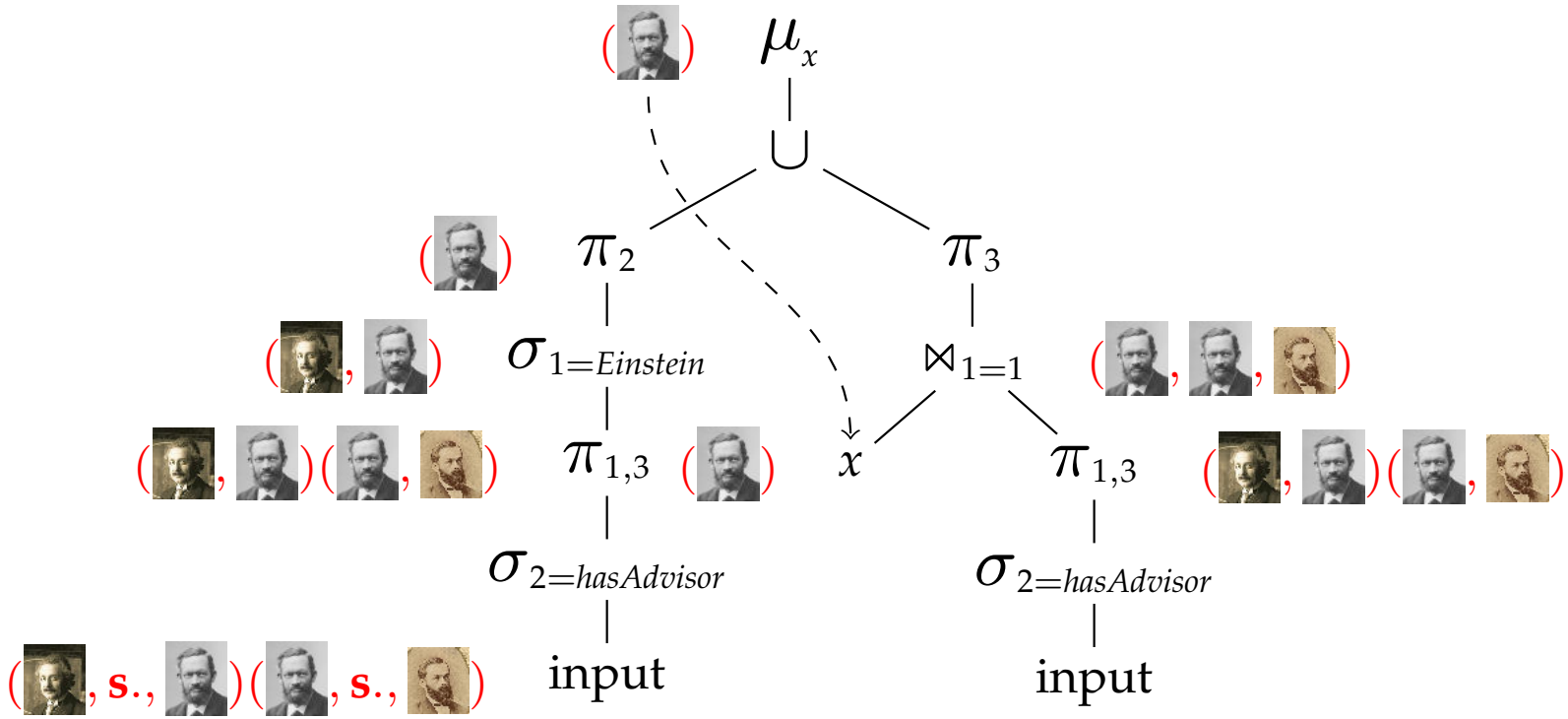
Conclusion

Answering Queries with Unix Shell: Approach


```
facts(X, Y, Z) :~ read_ntriples input

adv(X, Y) :- facts(X, "hasAdvisor", Y).

result(Y) :- adv("Einstein", Y).
result(Y) :- result(X), adv(X, Y).
```

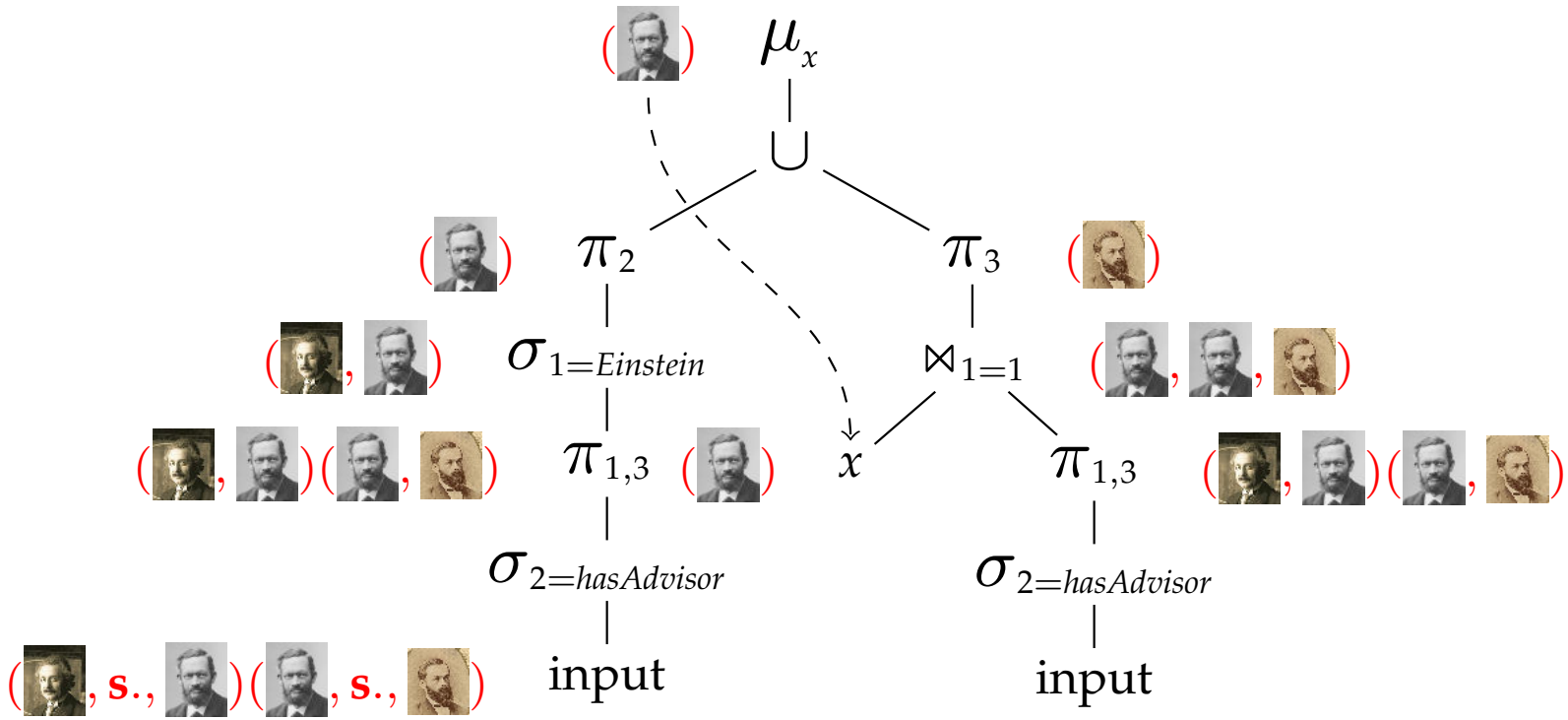


Answering Queries with Unix Shell: Approach


```
facts(X, Y, Z) :~ read_ntriples input

adv(X, Y) :- facts(X, "hasAdvisor", Y).

result(Y) :- adv("Einstein", Y).
result(Y) :- result(X), adv(X, Y).
```

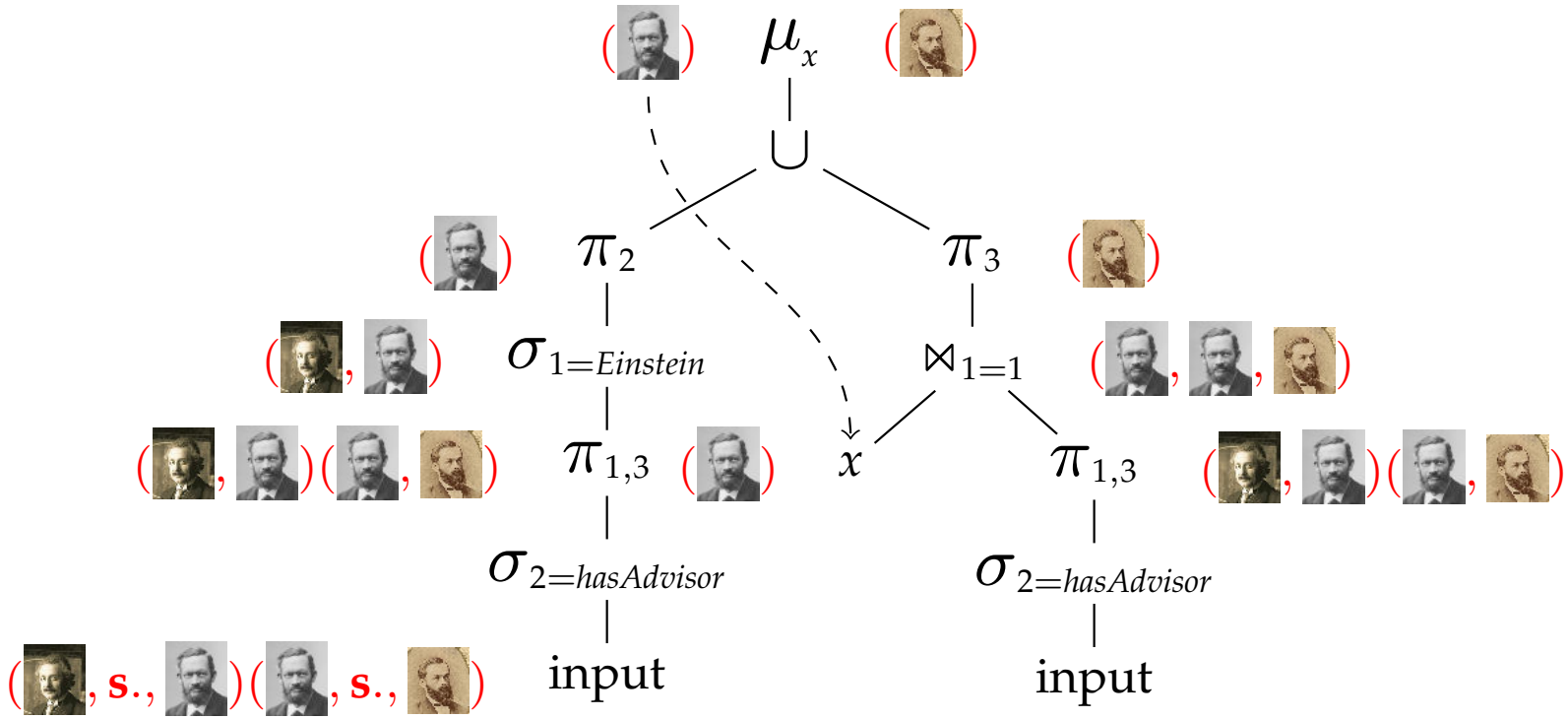


Answering Queries with Unix Shell: Approach


```
facts(X, Y, Z) :~ read_ntriples input

adv(X, Y) :- facts(X, "hasAdvisor", Y).

result(Y) :- adv("Einstein", Y).
result(Y) :- result(X), adv(X, Y).
```

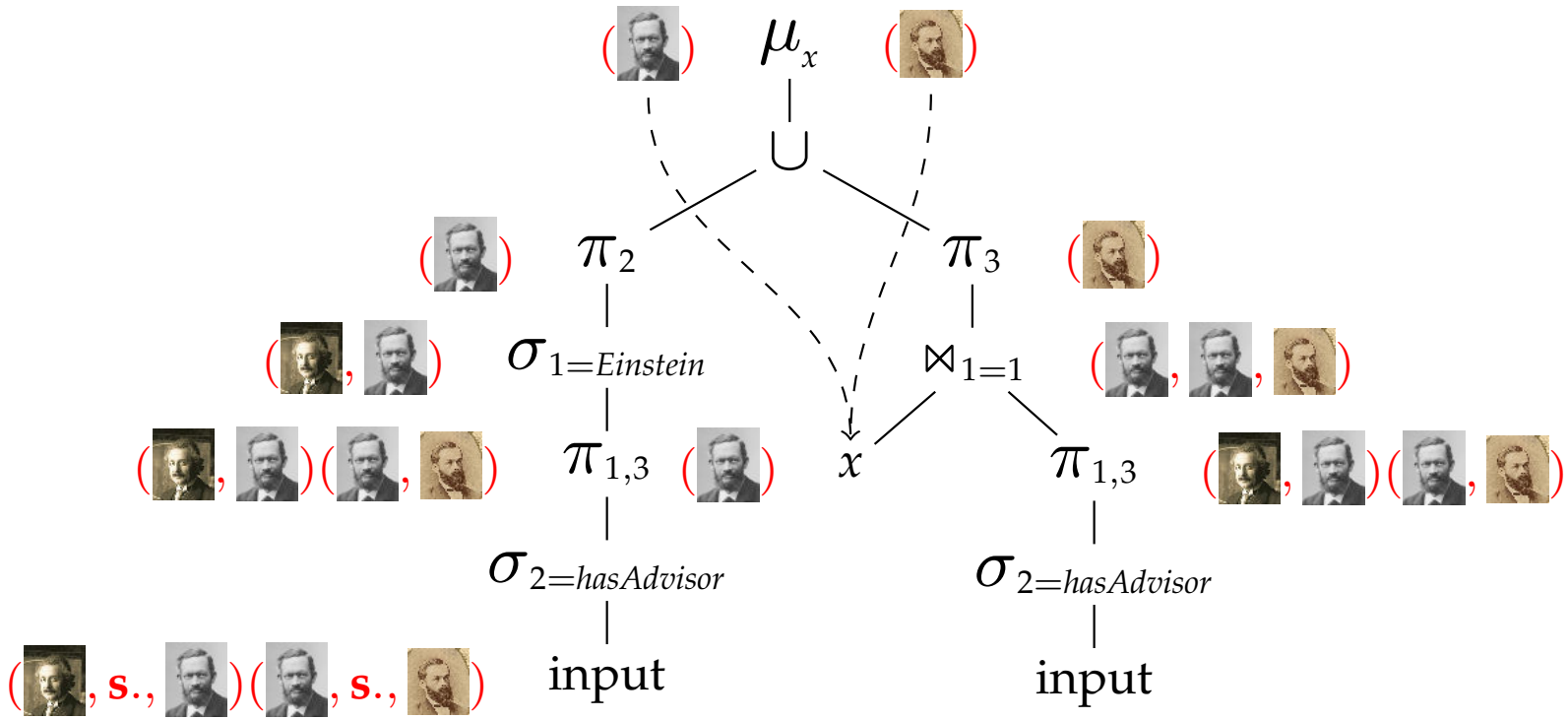


Answering Queries with Unix Shell: Approach


```
facts(X, Y, Z) :~ read_ntriples input

adv(X, Y) :- facts(X, "hasAdvisor", Y).

result(Y) :- adv("Einstein", Y).
result(Y) :- result(X), adv(X, Y).
```



Expanding the YAGO knowledge base

Rebele

The YAGO knowledge base

Outline

Using YAGO for the humanities

Adding Words to Regexes

Answering Queries with Unix Shell

Motivation

Idea

Approach

Optimization

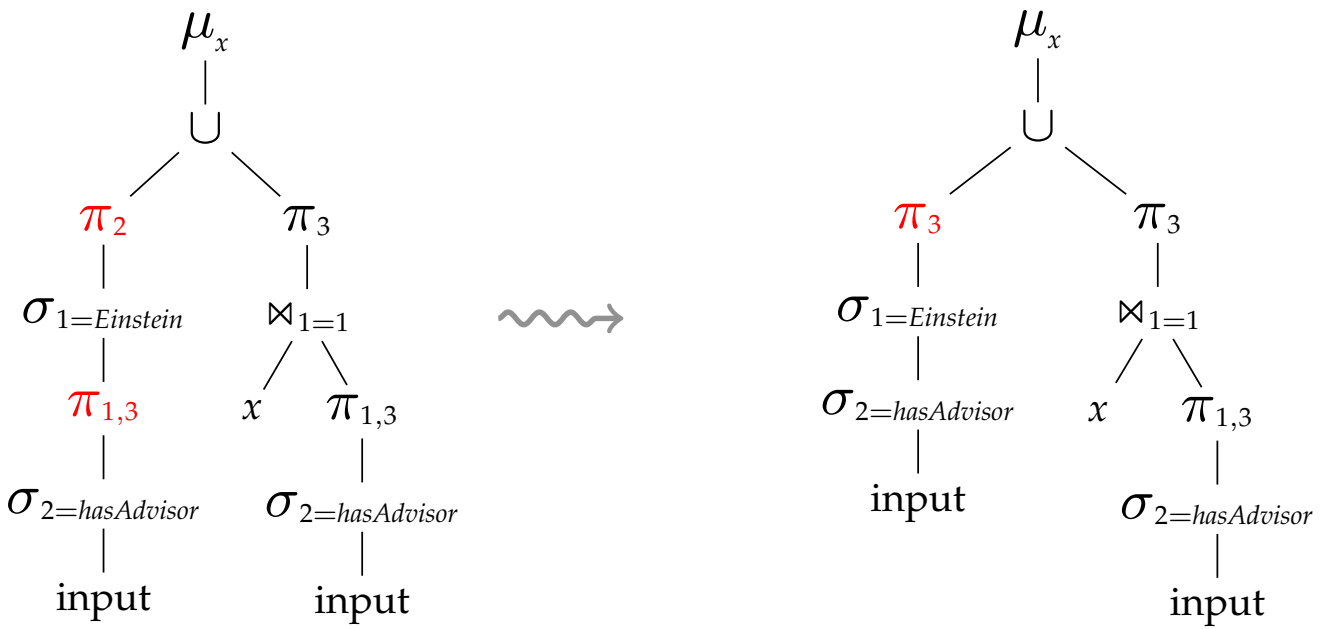
Experiments

Summary

Conclusion

Answering Queries with Unix Shell: Approach

Optimizations:



Expanding the YAGO knowledge base

Rebele

The YAGO knowledge base

Outline

Using YAGO for the humanities

Adding Words to Regexes

Answering Queries with Unix Shell

Motivation

Idea

Approach

Optimization

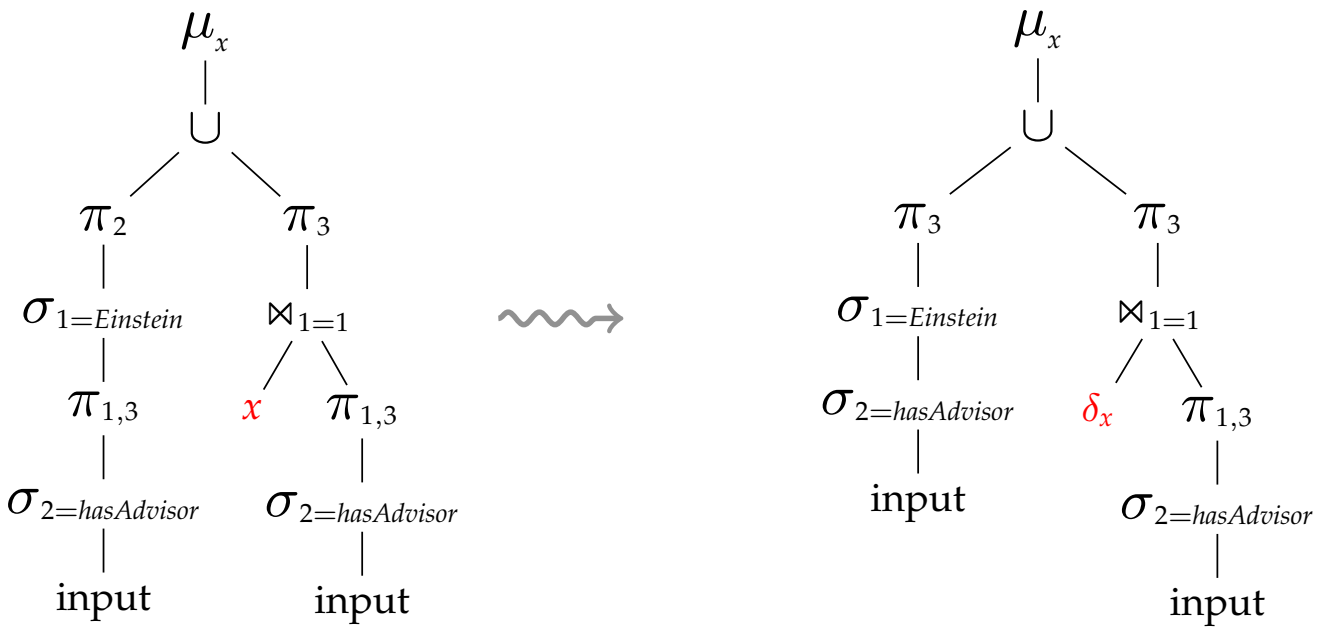
Experiments

Summary

Conclusion

Answering Queries with Unix Shell: Approach

Optimizations:



Expanding the YAGO knowledge base

Rebele

The YAGO knowledge base

Outline

Using YAGO for the humanities

Adding Words to Regexes

Answering Queries with Unix Shell

Motivation

Idea

Approach

Optimization

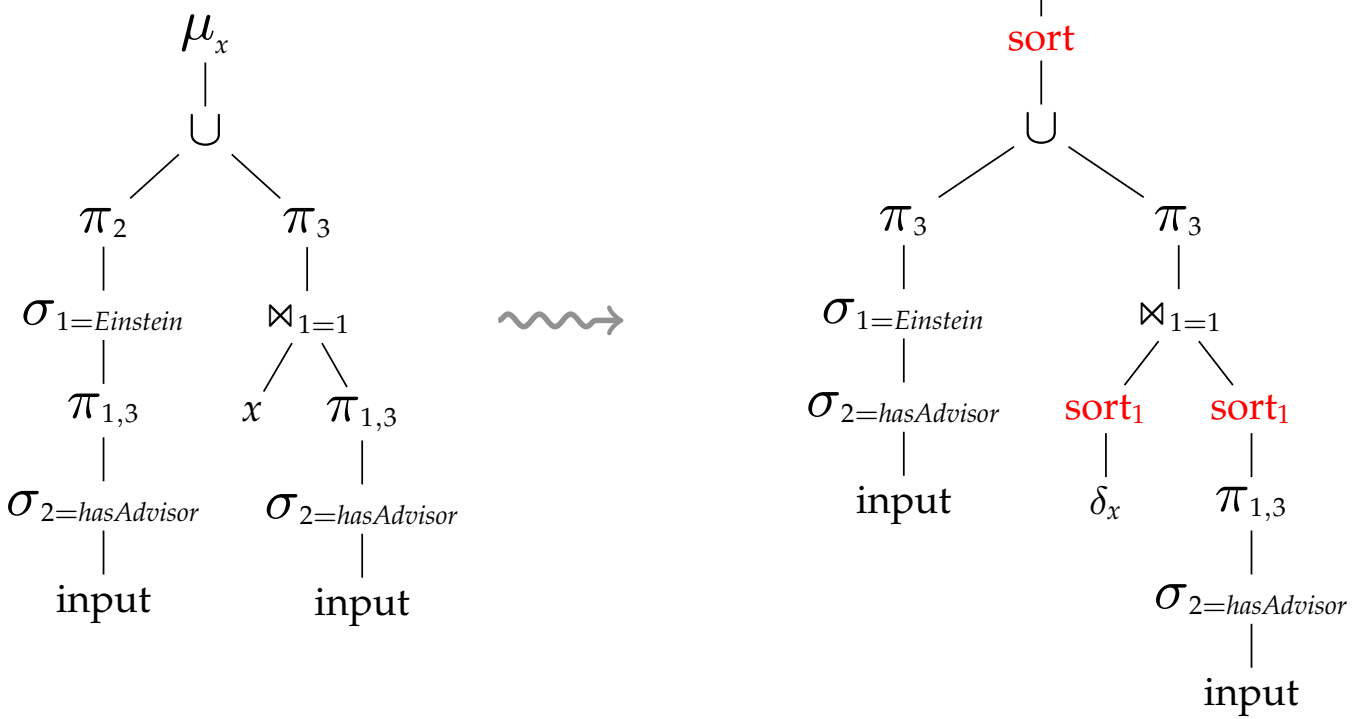
Experiments

Summary

Conclusion

Answering Queries with Unix Shell: Approach

Optimizations:



Expanding the YAGO knowledge base

Rebele

The YAGO knowledge base

Outline

Using YAGO for the humanities

Adding Words to Regexes

Answering Queries with Unix Shell

Motivation

Idea

Approach

Optimization

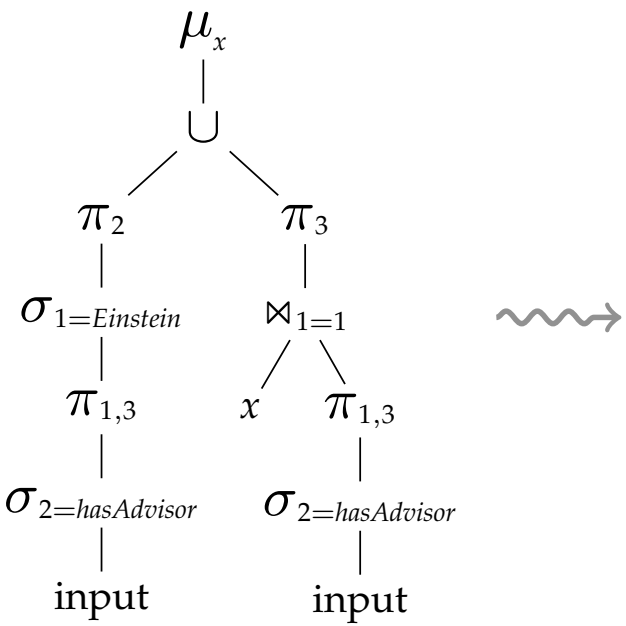
Experiments

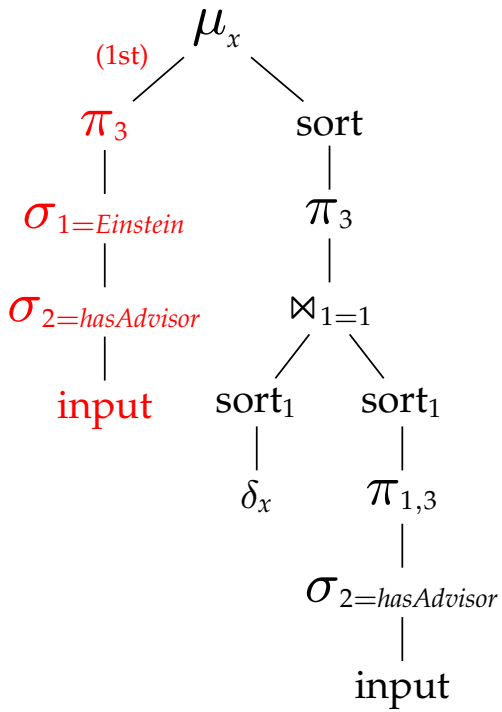
Summary

Conclusion

Answering Queries with Unix Shell: Approach

Optimizations:





Expanding the YAGO knowledge base

Rebele

The YAGO knowledge base

Outline

Using YAGO for the humanities

Adding Words to Regexes

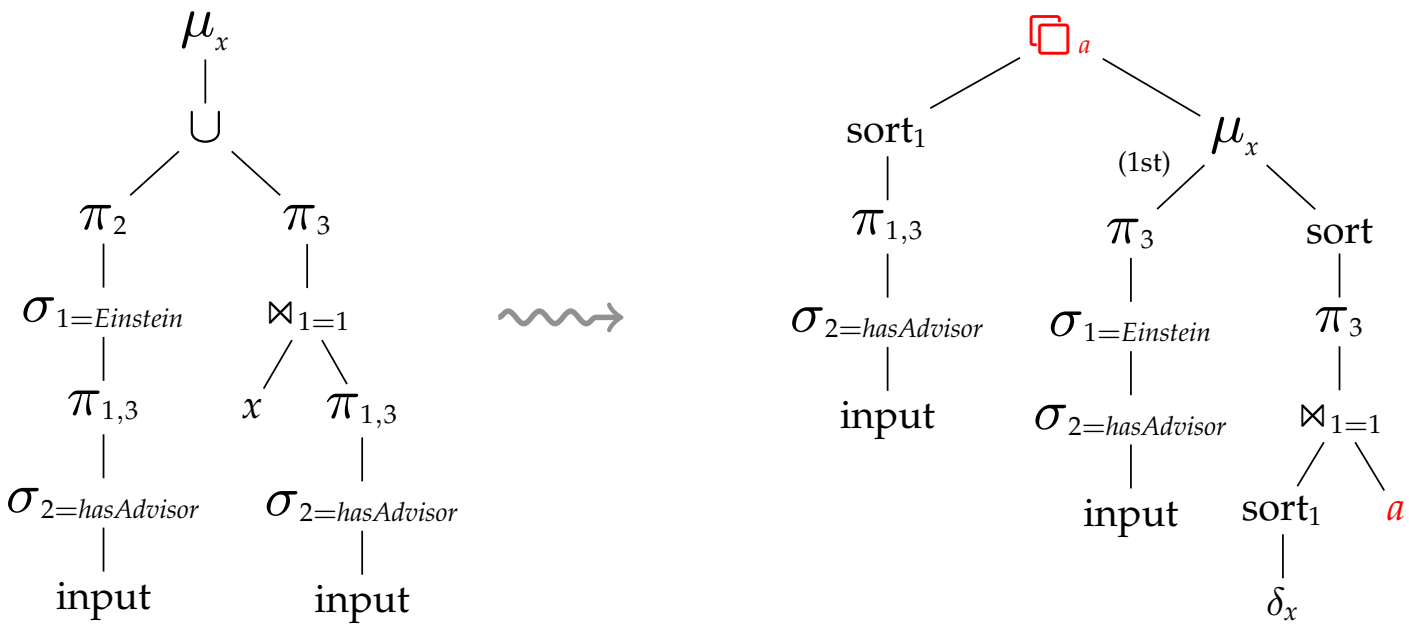
Answering Queries with Unix Shell

- Motivation
- Idea
- Approach
- Optimization
- Experiments
- Summary

Conclusion

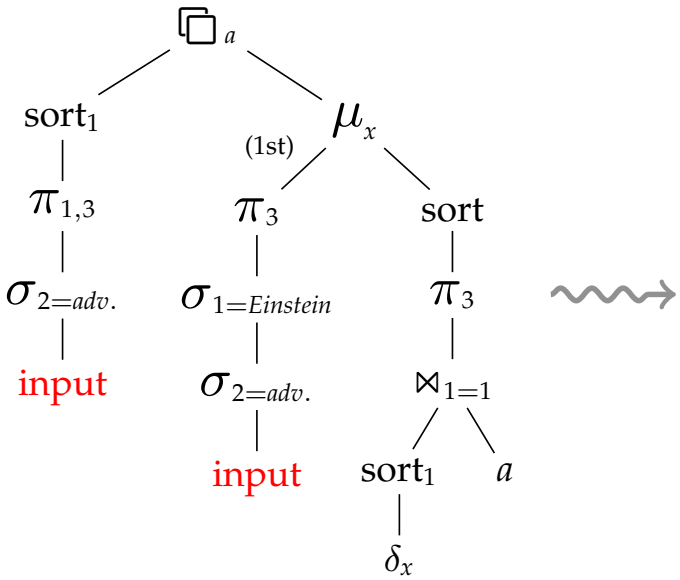
Answering Queries with Unix Shell: Approach

Optimizations:



Expanding the YAGO knowledge base
Rebele
The YAGO knowledge base
Outline
Using YAGO for the humanities
Adding Words to Regexes
Answering Queries with Unix Shell
Motivation
Idea
Approach
Optimization
Experiments
Summary
Conclusion

Answering Queries with Unix Shell: Approach

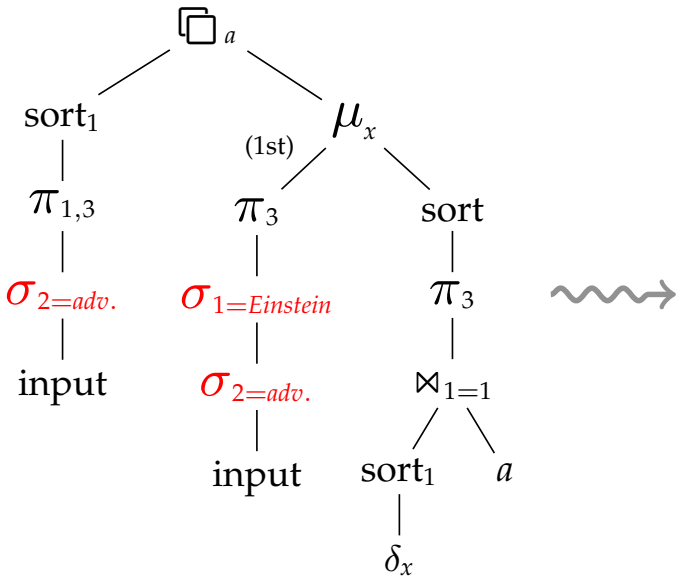



```
awk '
 (    $1 == "Einstein"
   && $2 == "hasAdvisor")
    { print $3 >> "b" }
 ($2 == "hasAdvisor")
    { print $1 FS $3 >> "pre_a" }
  ' <(read_ntriples input)

# lock a
(
    sort -k 1 pre_a > a
    # unlock a
) &
```


Expanding the YAGO knowledge base
Rebele
The YAGO knowledge base
Outline
Using YAGO for the humanities
Adding Words to Regexes
Answering Queries with Unix Shell
Motivation
Idea
Approach
Optimization
Experiments
Summary
Conclusion

Answering Queries with Unix Shell: Approach

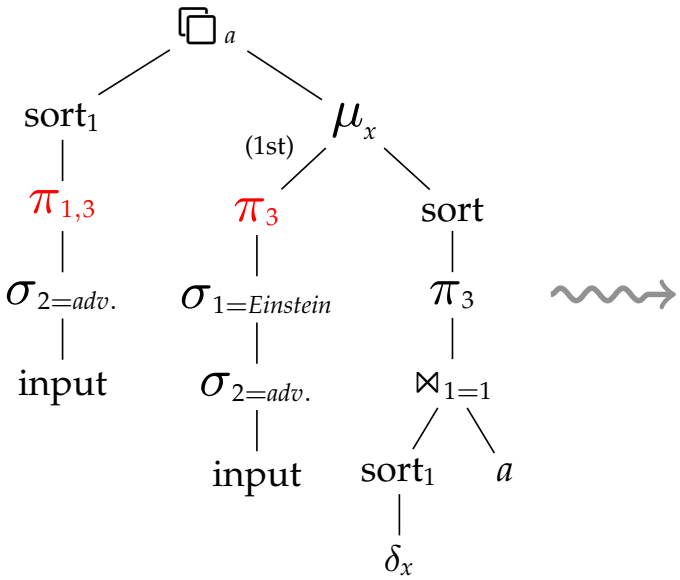



```
awk '
 (    $1 == "Einstein"
   && $2 == "hasAdvisor")
    { print $3 >> "b" }
 ($2 == "hasAdvisor")
    { print $1 FS $3 >> "pre_a" }
  ' <(read_ntriples input)

# lock a
(
    sort -k 1 pre_a > a
    # unlock a
) &
```


Expanding the YAGO knowledge base
Rebele
The YAGO knowledge base
Outline
Using YAGO for the humanities
Adding Words to Regexes
Answering Queries with Unix Shell
Motivation
Idea
Approach
Optimization
Experiments
Summary
Conclusion

Answering Queries with Unix Shell: Approach

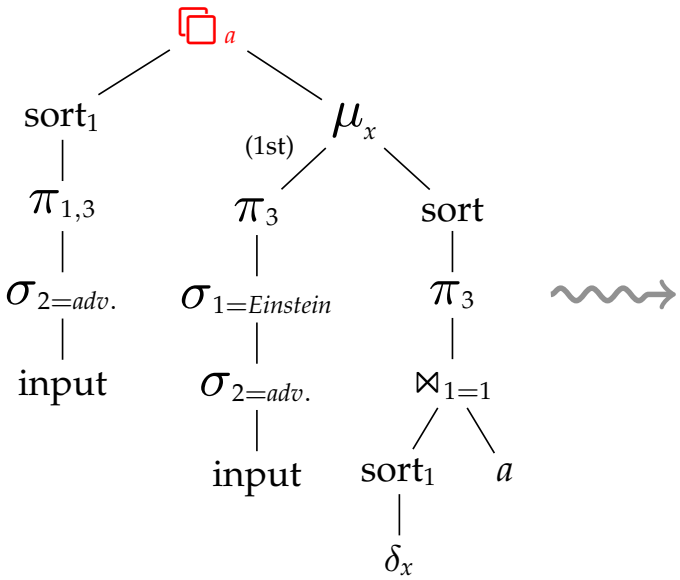



```
awk '
 (    $1 == "Einstein"
   && $2 == "hasAdvisor")
    { print $3 >> "b" }
 ($2 == "hasAdvisor")
    { print $1 FS $3 >> "pre_a" }
  ' <(read_ntriples input)

# lock a
(
    sort -k 1 pre_a > a
    # unlock a
) &
```


Expanding the YAGO knowledge base
Rebele
The YAGO knowledge base
Outline
Using YAGO for the humanities
Adding Words to Regexes
Answering Queries with Unix Shell
Motivation
Idea
Approach
Optimization
Experiments
Summary
Conclusion

Answering Queries with Unix Shell: Approach

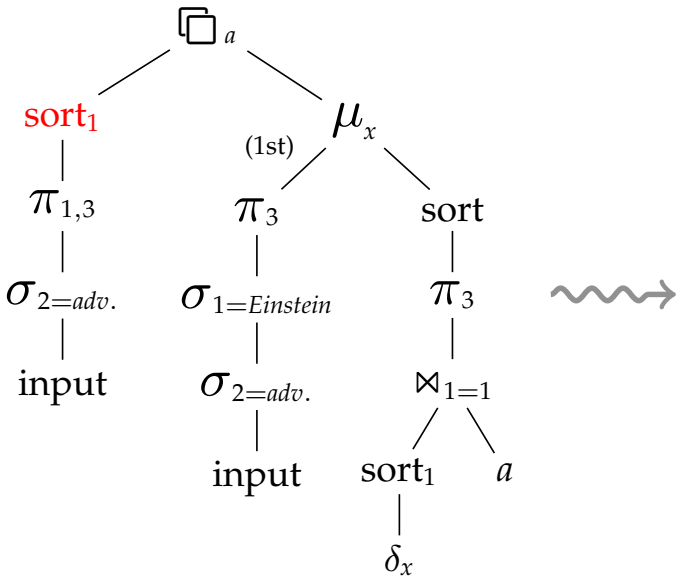



```
awk '
 (    $1 == "Einstein"
   && $2 == "hasAdvisor")
    { print $3 >> "b" }
 ($2 == "hasAdvisor")
    { print $1 FS $3 >> "pre_a" }
  ' <(read_ntriples input)

# lock a
(
    sort -k 1 pre_a > a
    # unlock a
) &
```


Expanding the YAGO knowledge base
Rebele
The YAGO knowledge base
Outline
Using YAGO for the humanities
Adding Words to Regexes
Answering Queries with Unix Shell
Motivation
Idea
Approach
Optimization
Experiments
Summary
Conclusion

Answering Queries with Unix Shell: Approach

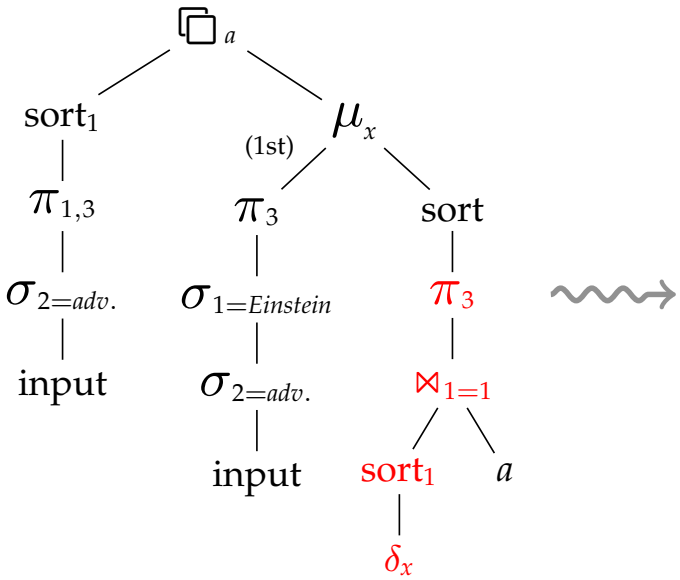



```
awk '
 (    $1 == "Einstein"
   && $2 == "hasAdvisor")
    { print $3 >> "b" }
 ($2 == "hasAdvisor")
    { print $1 FS $3 >> "pre_a" }
  ' <(read_ntriples input)

# lock a
(
    sort -k 1 pre_a > a
    # unlock a
) &
```


Expanding the YAGO knowledge base
Rebele
The YAGO knowledge base
Outline
Using YAGO for the humanities
Adding Words to Regexes
Answering Queries with Unix Shell
Motivation
Idea
Approach
Optimization
Experiments
Summary
Conclusion

Answering Queries with Unix Shell: Approach

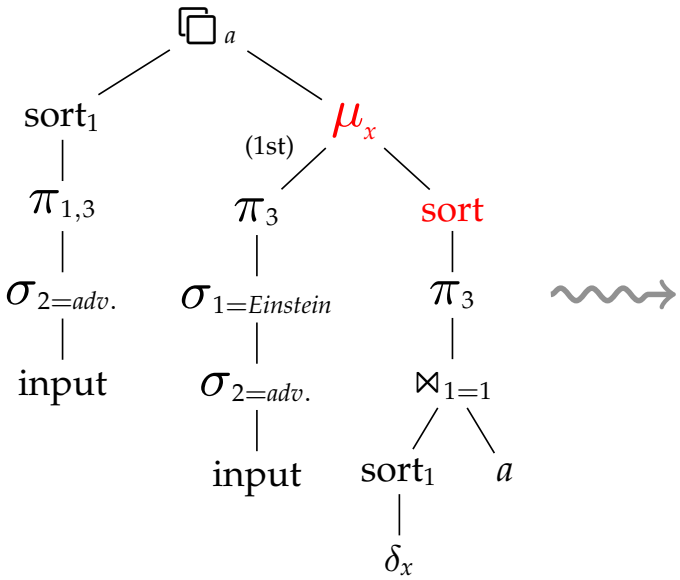



```
while
# ...
sort -k 1 -u
    <( # wait for a
        join -1 1 -2 1 -o 2.2
          <(sort -k 1 -u delta)
          a
     )

# ...
[ -s delta ];
do continue; done
```


Expanding the YAGO knowledge base
Rebele
The YAGO knowledge base
Outline
Using YAGO for the humanities
Adding Words to Regexes
Answering Queries with Unix Shell
Motivation
Idea
Approach
Optimization
Experiments
Summary
Conclusion

Answering Queries with Unix Shell: Approach

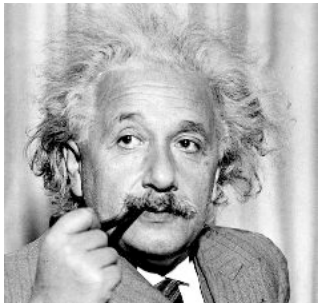



```
while
# ...
sort -k 1 -u
    <( # wait for a
        join -1 1 -2 1 -o 2.2
          <(sort -k 1 -u delta)
          a
     )

# ...
[ -s delta ];
do continue; done
```


Expanding the YAGO knowledge base
Rebele
The YAGO knowledge base
Outline
Using YAGO for the humanities
Adding Words to Regexes
Answering Queries with Unix Shell
Motivation
Idea
Approach
Optimization
Experiments
Summary
Conclusion

Answering Queries with Unix Shell: Optimization

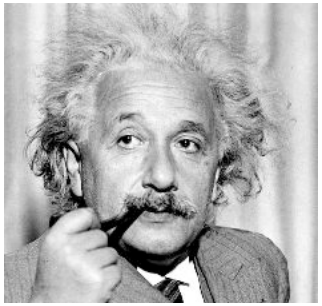


How can I find all professors?

```
Professor(X) :- Person(X),
                teachesCourse(X,Y).
Professor(X) :- advisorOf(X,Y),
                Professor(Y).

Person(X) :- Employee(X).
Person(X) :- Professor(X).
```

Answering Queries with Unix Shell: Optimization

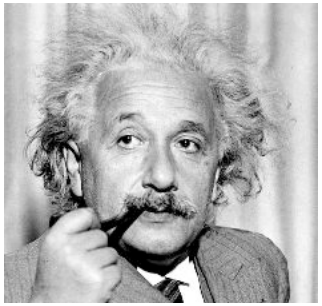


How can I find all professors?

```
Professor(X) :- Person(X),
                teachesCourse(X,Y).
Professor(X) :- advisorOf(X,Y),
                Professor(Y).

Person(X) :- Employee(X).
Person(X) :- Professor(X).
```

Answering Queries with Unix Shell: Optimization



How can I find all professors?

```
Professor(X) :- Person(X),
                teachesCourse(X,Y).
Professor(X) :- advisorOf(X,Y),
                Professor(Y).

Person(X) :- Employee(X).
Person(X) :- Professor(X).
```

Combining the first and the last rule leads to

```
Professor(X) :- Professor(X),
                teachesCourse(X,Y).
```

The YAGO knowledge base

Outline

Using YAGO for the humanities

Adding Words to Regexes

Answering Queries with Unix Shell

Motivation

Idea

Approach

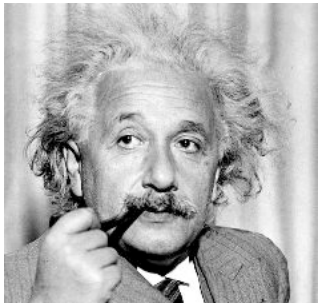
Optimization

Experiments

Summary

Conclusion

Answering Queries with Unix Shell: Optimization



How can I find all professors?

```
Professor(X) :- Person(X),
                teachesCourse(X,Y).
Professor(X) :- advisorOf(X,Y),
                Professor(Y).

Person(X) :- Employee(X).
Person(X) :- Professor(X).
```

Combining the first and the last rule leads to

```
Professor(X) :- Professor(X),
                teachesCourse(X,Y).
```

The YAGO knowledge base

Outline

Using YAGO for the humanities

Adding Words to Regexes

Answering Queries with Unix Shell

Motivation

Idea

Approach

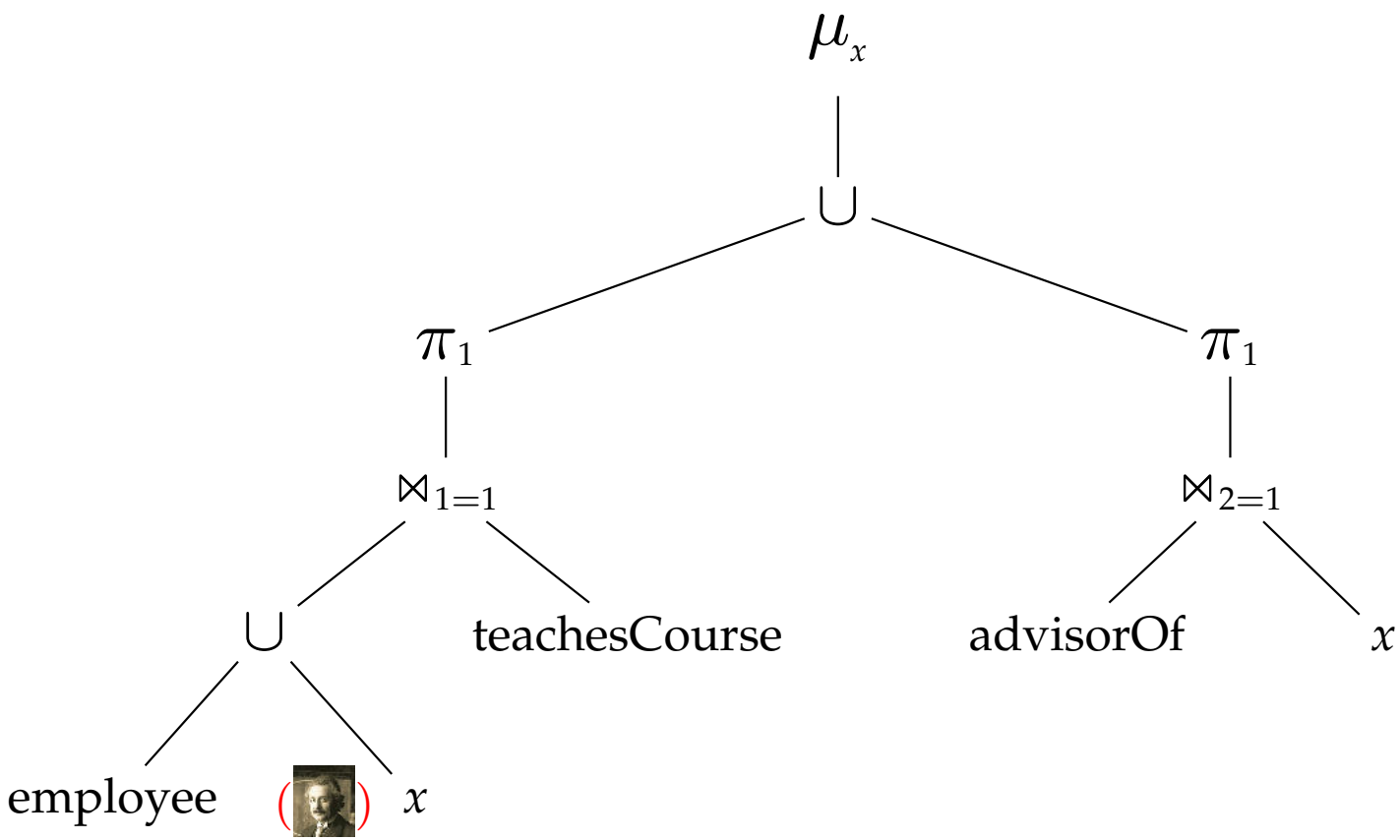
Optimization

Experiments

Summary

Conclusion

Answering Queries with Unix Shell: Optimization



Expanding the YAGO knowledge base

Rebele

The YAGO knowledge base

Outline

Using YAGO for the humanities

Adding Words to Regexes

Answering Queries with Unix Shell

Motivation

Idea

Approach

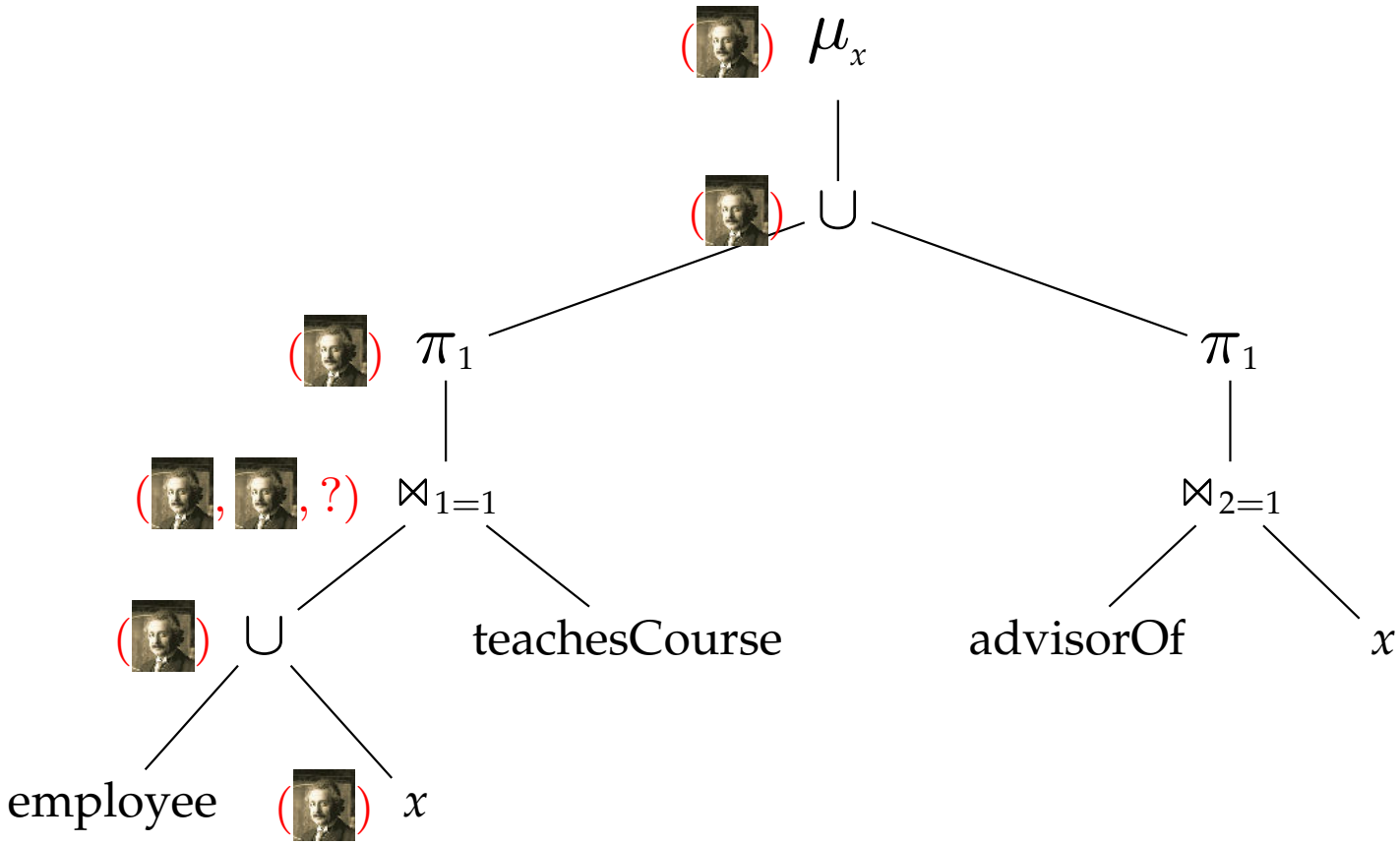
Optimization

Experiments

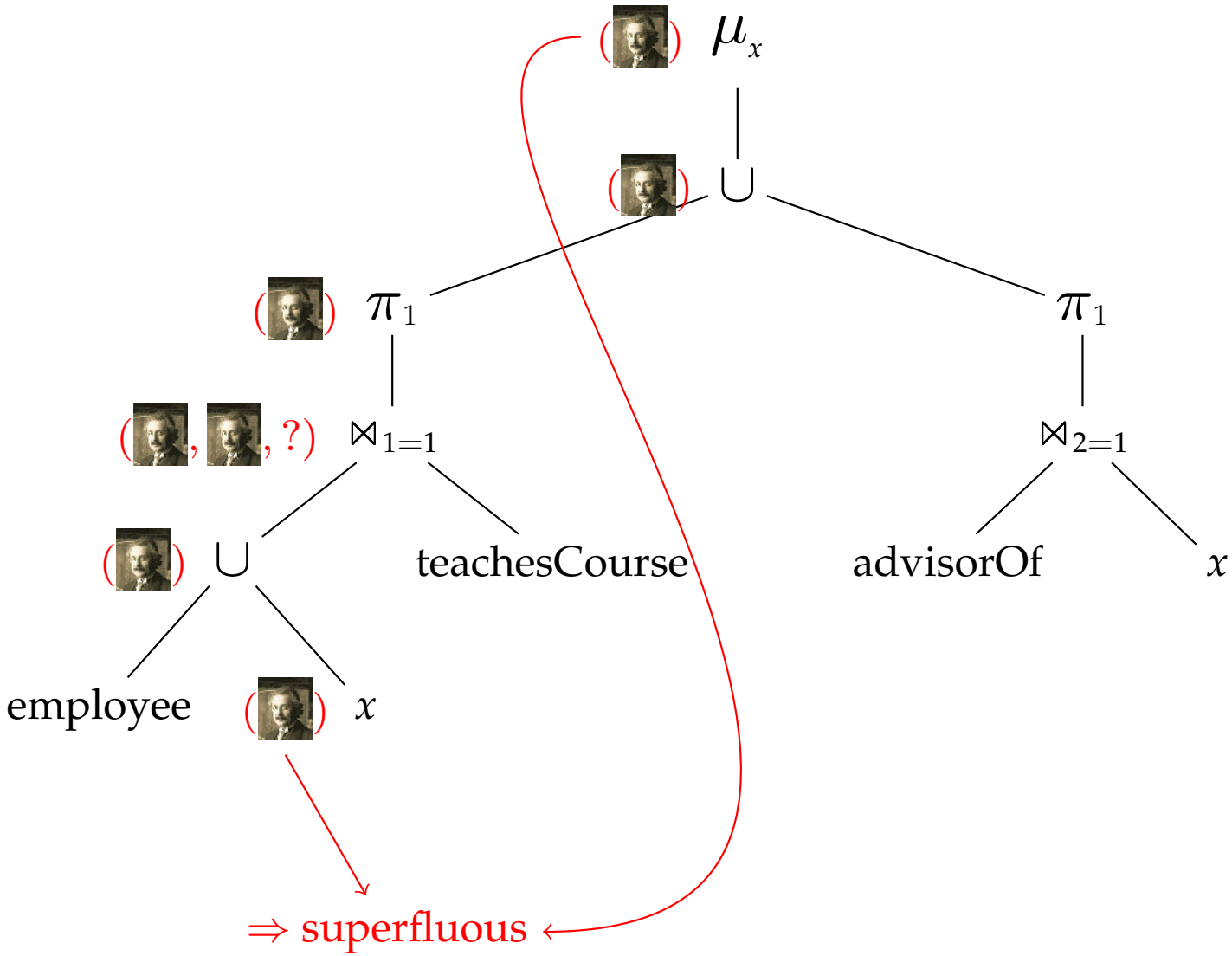
Summary

Conclusion

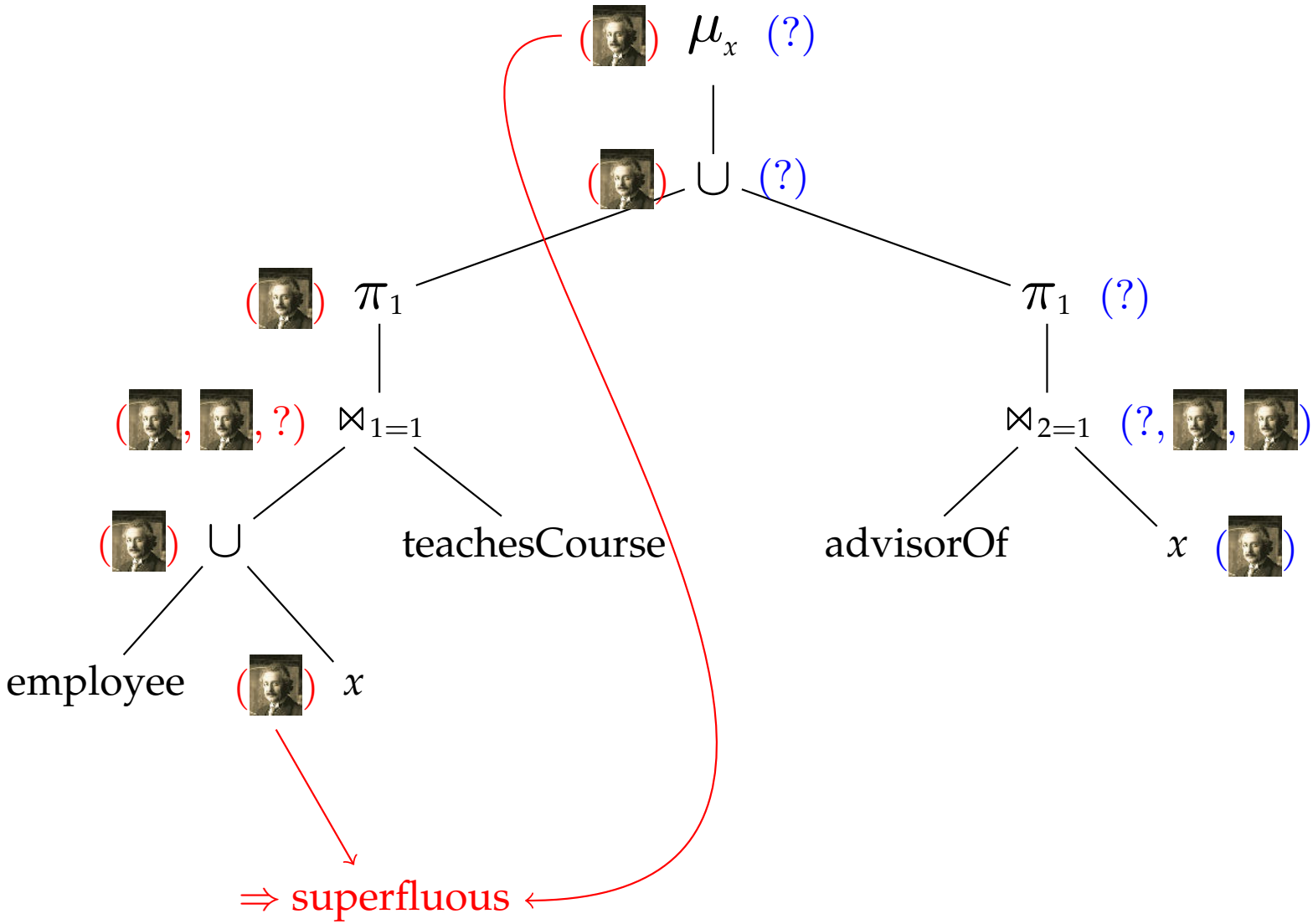
Answering Queries with Unix Shell: Optimization



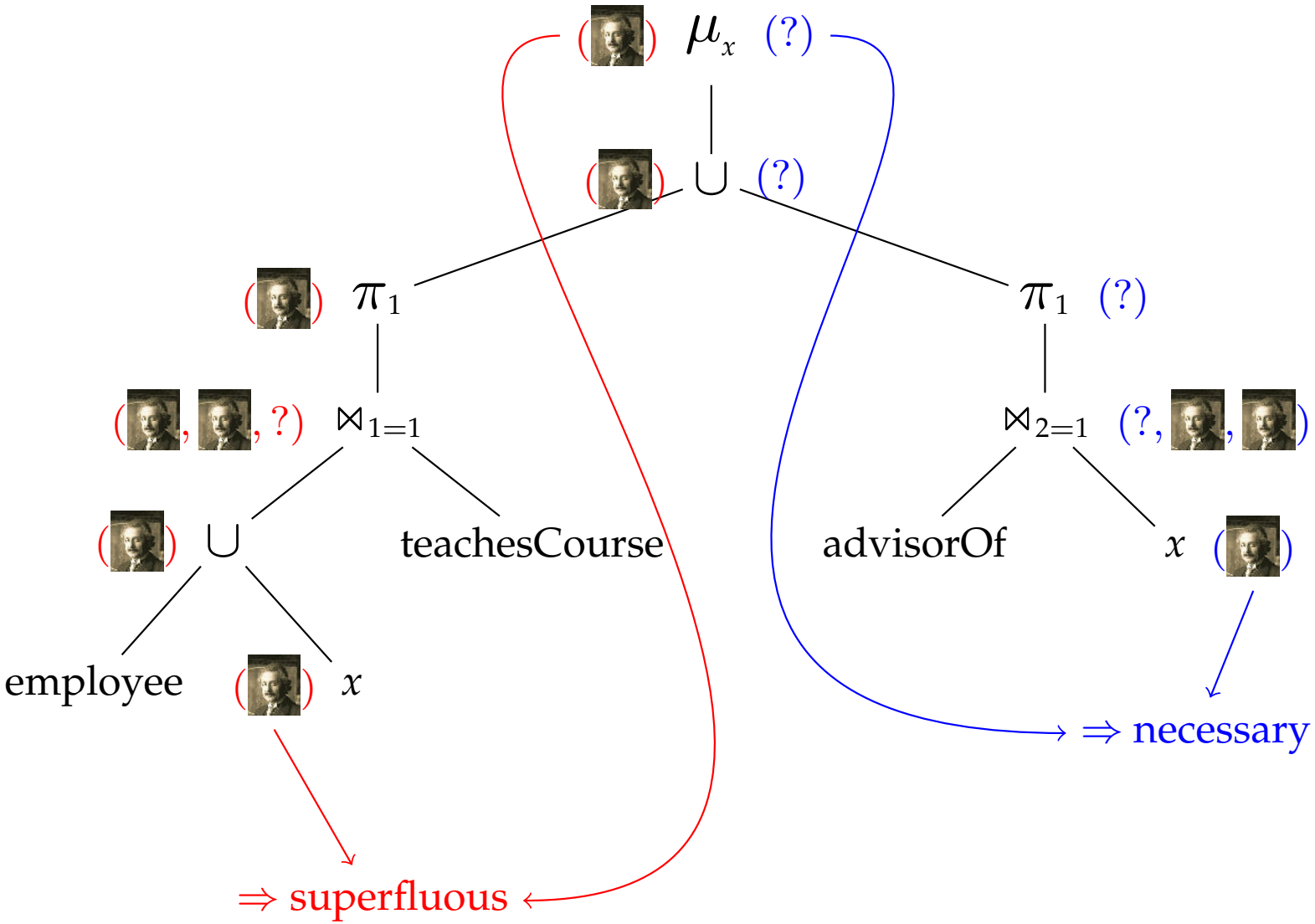
Answering Queries with Unix Shell: Optimization



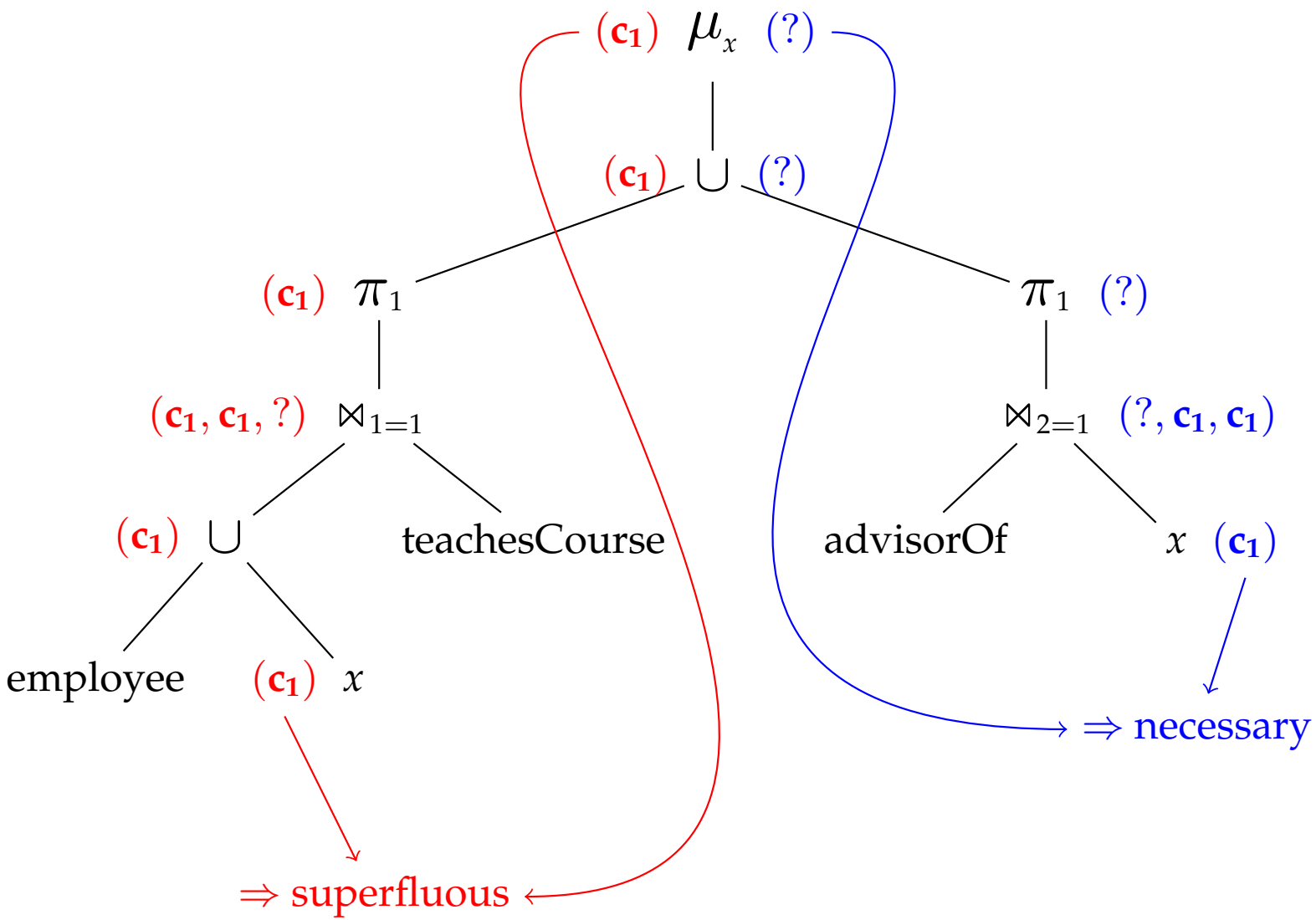
Answering Queries with Unix Shell: Optimization



Answering Queries with Unix Shell: Optimization



Answering Queries with Unix Shell: Optimization

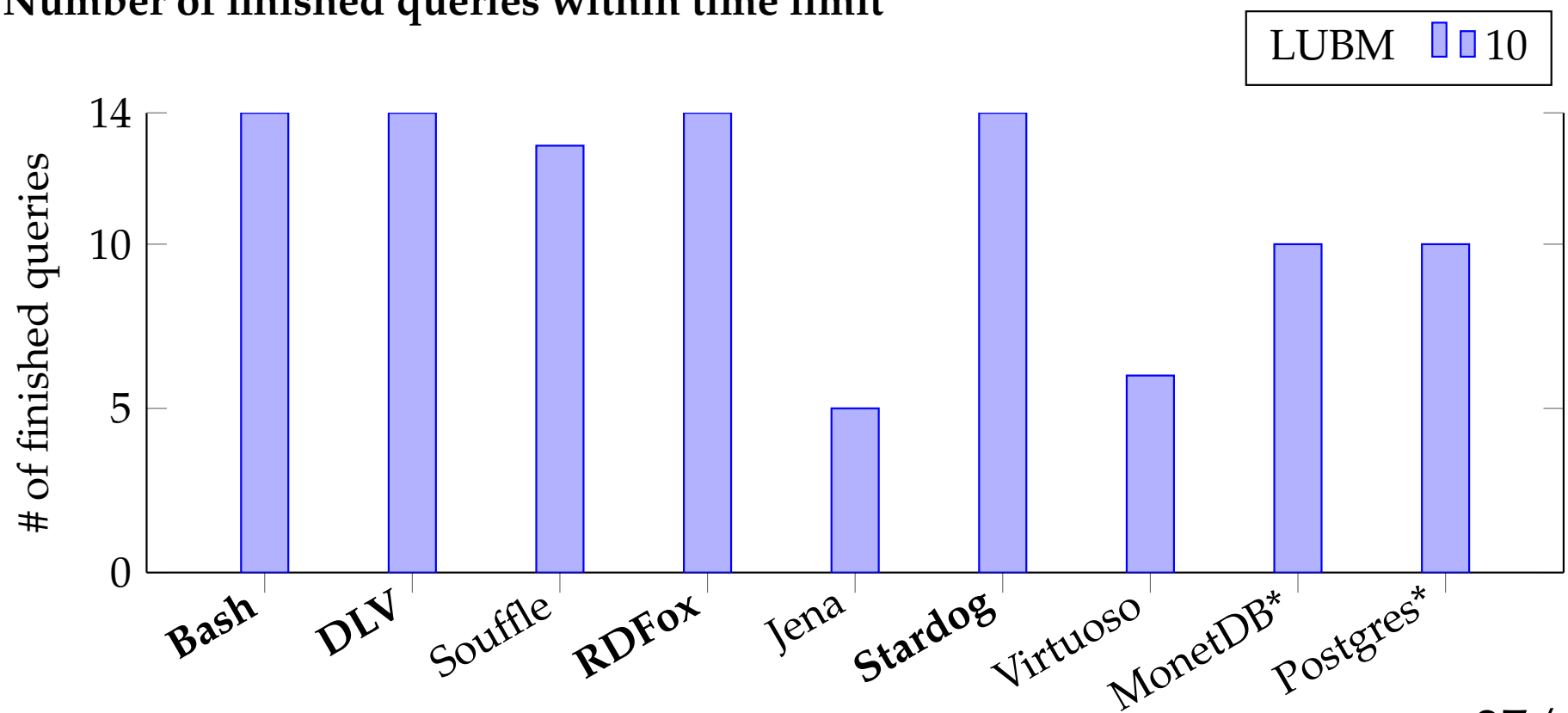


Expanding the YAGO knowledge base
Rebele
The YAGO knowledge base
Outline
Using YAGO for the humanities
Adding Words to Regexes
Answering Queries with Unix Shell
Motivation
Idea
Approach
Optimization
Experiments
Summary
Conclusion

Answering Queries with Unix Shell: Experiments

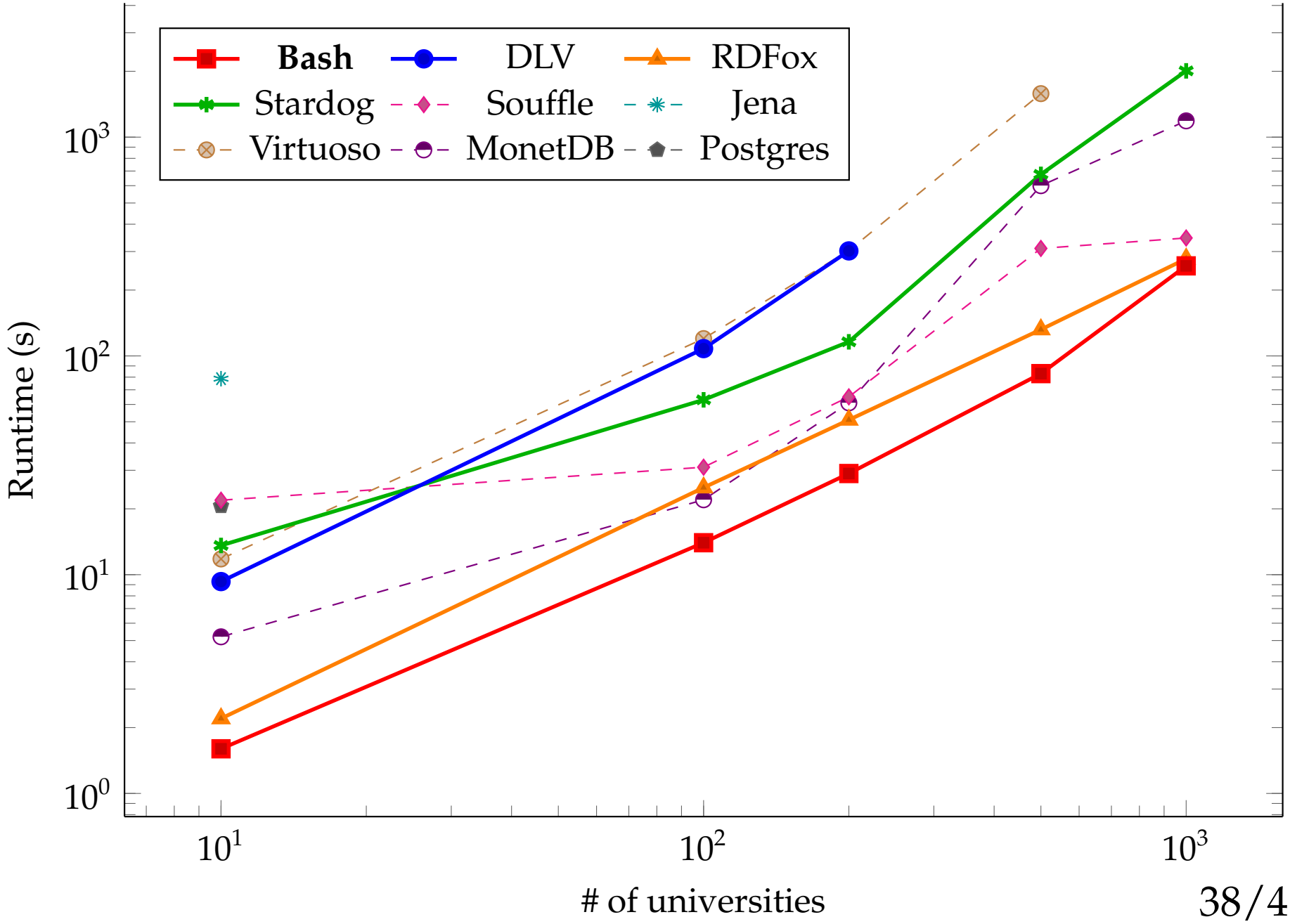
- Dataset: LUBM university benchmark
- 14 different queries
- Competitors: Datalog-based systems (DLV, Souffle, RDFox),
  Triple stores (Jena, Stardog, Virtuoso),
  Database management systems (MonetDB, Postgres)

Number of finished queries within time limit



* = we folded the TBox into the query

Answering Queries with Unix Shell: Experiments



Answering Queries with Unix Shell: Experiments

dataset	Bash	RDFox	BigDatalog	Stardog	Virtuoso
LiveJournal	117	**70**	532	941	-
orkut	225	**121**	1838	1123	-
friendster	**16306**	-	-	-	-

Table: Runtime for the reachability query, in seconds.

Expanding the YAGO knowledge base
Rebele
The YAGO knowledge base
Outline
Using YAGO for the humanities
Adding Words to Regexes
Answering Queries with Unix Shell
Motivation
Idea
Approach
Optimization
Experiments
Summary
Conclusion

Answering Queries with Unix Shell: Summary

Summary:

- Preprocess large datasets without installing software
- Supports subset of SPARQL / OWL and Datalog as query language
- Try it online at `https://www.thomasrebele.org/projects/bashlog`
- Source code available at `https://github.com/thomasrebele/bashlog`

Future work:

- Numerical comparisons
- Aggregations (e.g., max, count)

Publication: ISWC 2018 (full paper)


Thomas Rebele


Thomas P. Tanon


Fabian Suchanek

The YAGO knowledge base
Outline
Using YAGO for the humanities
Adding Words to Regexes
Answering Queries with Unix Shell
Motivation
Idea
Approach
Optimization
Experiments
Summary
Conclusion

Conclusion

This thesis showed how to extend YAGO along several axes:

- Improve completeness w.r.t. people
- Automatically repairing of its regular expressions
- Preprocessing queries using only a Bash shell

Other accomplishments:

- Source code of all contributions is available online
- Publications at ISWC 2016 (resource paper), ISWC 2017 (demo, workshop), PAKDD 2018 (full paper), ISWC 2018 (full paper)

Future work:

- More studies on human society using facts from YAGO (ongoing)
- Combine YAGO and Wikidata
- Queries with numerical comparisons and aggregations