Information Extraction

Liat Peterfreund

Based also on slides by Benny Kimelfeld

What is Information Extraction?

data-in-text (unstructured) → **data-in-db (structured)**

"Information Extraction (IE) is the name given to any process which selectively **structures and combines data** which is found, explicitly stated or implied, in one or more texts. The final output of the extraction process varies; in every case, however, it can be transformed so as to **populate some type of database**."

J. Cowie and Y. Wilks.
Handbook of Natural Language Processing 2000

Where does it occur in real life?

Social-Media

Political campaigning, trends, …

(Cyber) Security

Terror recruit, child exploit, human trafficking, …

Life-Science

Biological and medical knowledge bases

Email Management

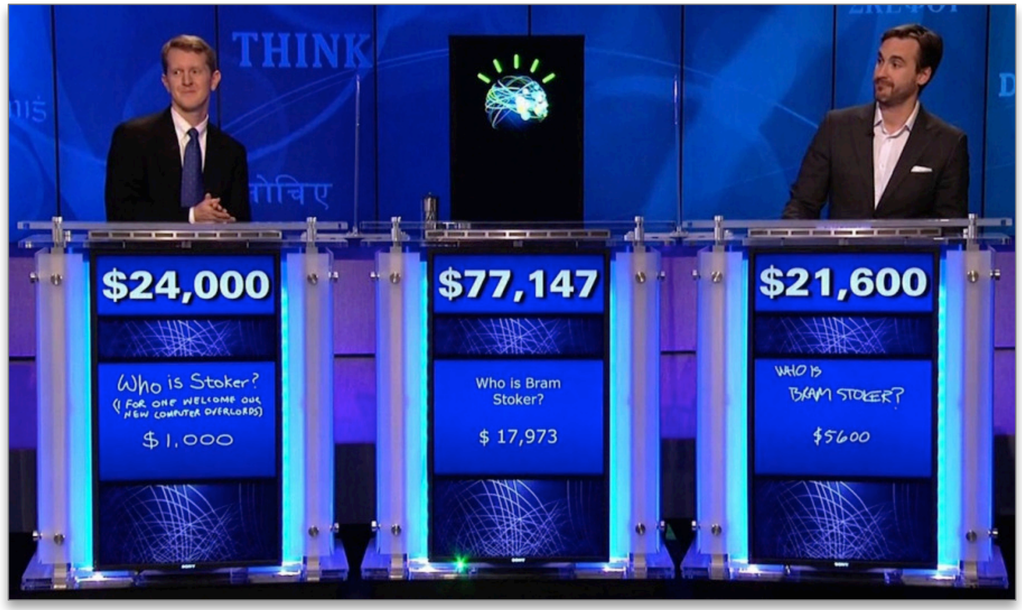
Auto-completion, span detector, content suggestion

Semantic Web

Domain specific and open domain knowledge bases

Examples from IBM Research

Jeopardy! challenge: Watson with public KBs (e.g., YAGO) + information extracted from text (e.g., Wikipedia, 1m books)



HEALTH CANCER

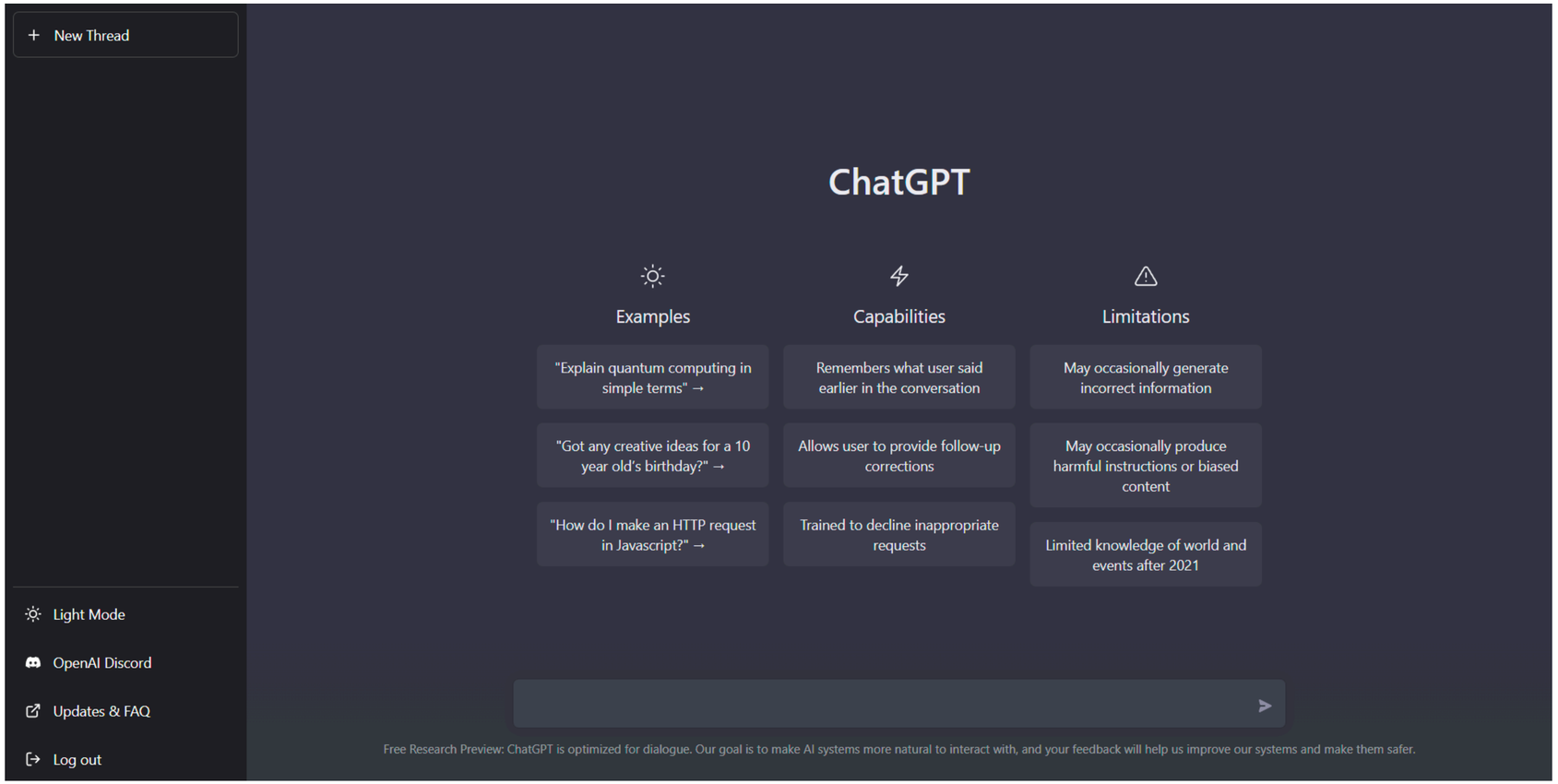
IBM Watson's Startling Cancer Coup

Bill Saporito @bilsap | Aug. 28, 2014

TIME

Extracted content from 70k MedLine papers towards insights on the tumour-suppressor p53 protein

Contemporary Example



Programming Paradigms for IE

Rule-base approaches: IBM's SystemT

```
create view Caps as
extract regex /[A-Z](\w|-)+/ on D.text as name from Document D;

create view Last as
extract dictionary LastGaz on D.text as name from Document D;

create view CapsLast as
select  CombineSpans(C.name, L.name) as name
from    Caps C, Last L
where   FollowsTok(C.name, L.name, 0, 0);

...

create view PersonAll as
    (select R.name from FirstLast R) union all ...
     ... union all (select R.name from CapsLast R);

create view Person as select * from PersonAll R
consolidate on R.name using 'ContainedWithin';

output view Person;
```

"Regex formulas"

regex + join w/ previous views

union

projection

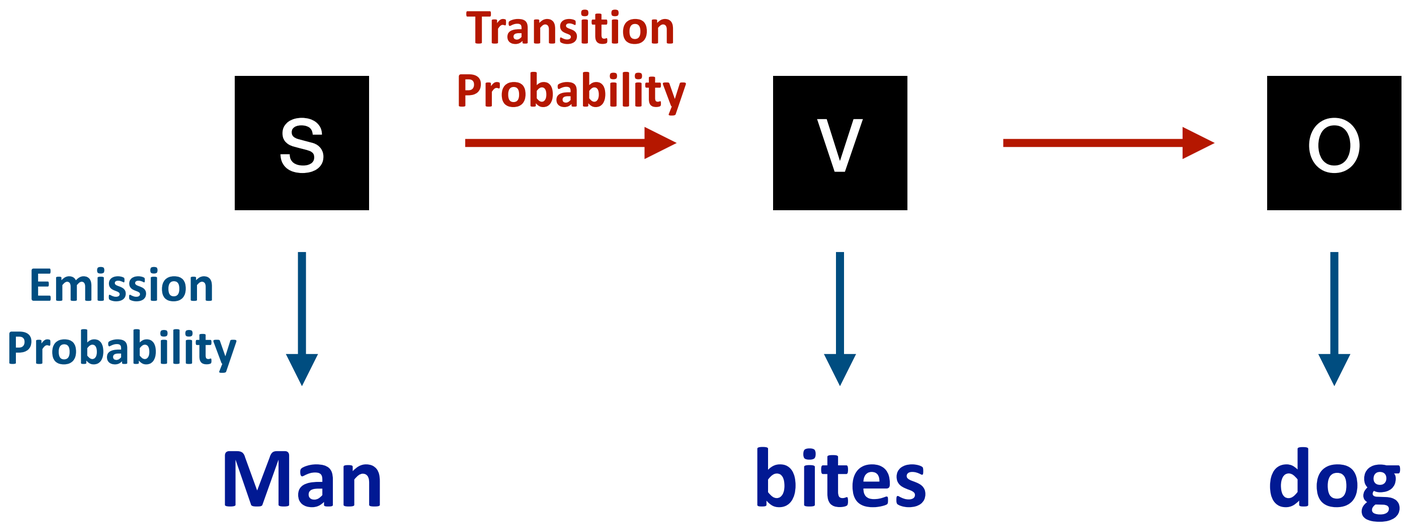
Cleaning

Base relations can also include NLP libs (e.g., Stanford's CoreNLP)
[cf KDD 2019 tutorial on SystemT]

- [Chiticariu, Krishnamurthy, Li, Raghavan, Reiss, Vaithyanathan, ACL 2010]

Statistical approaches: Hiden Markov Model

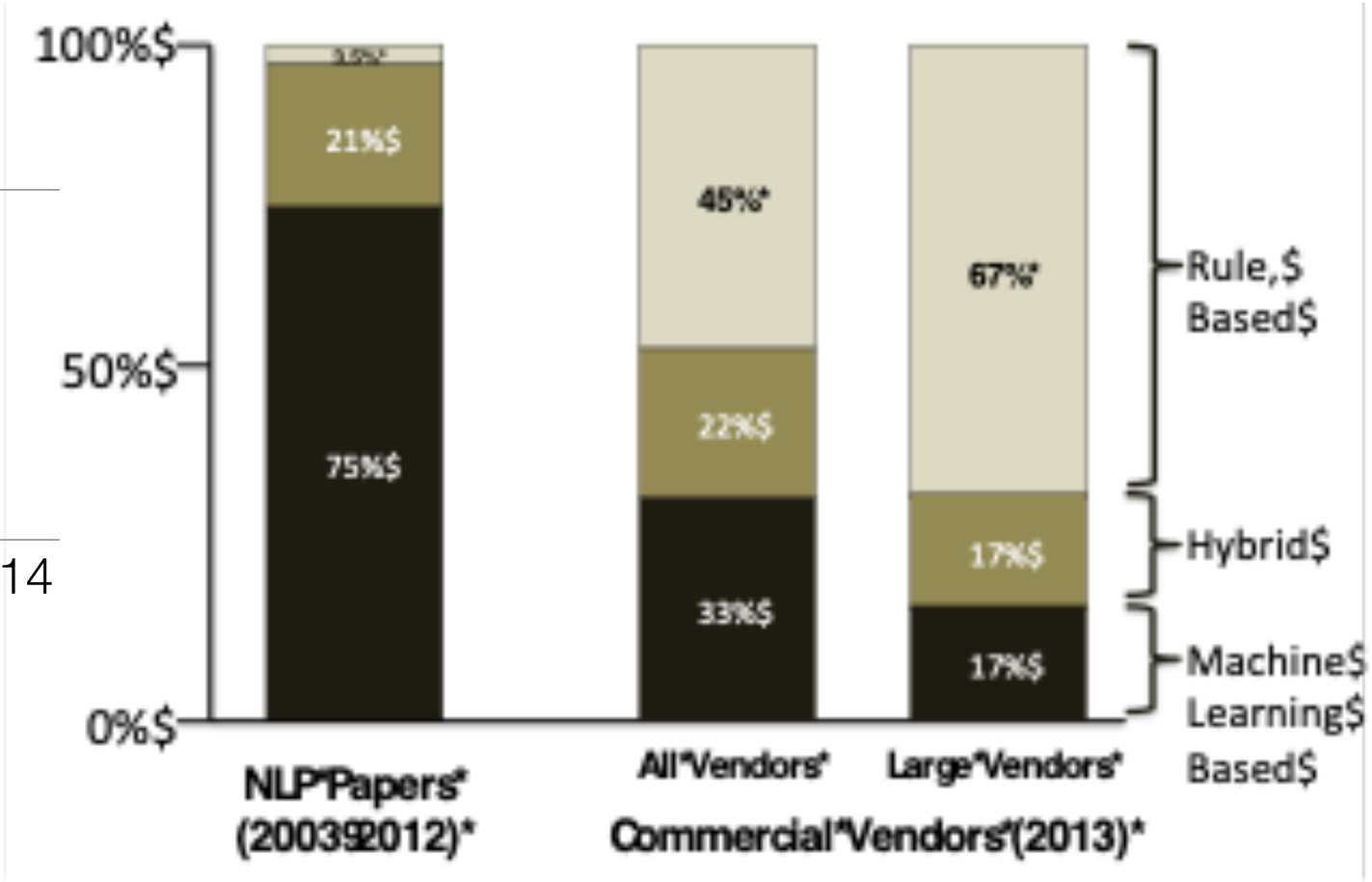
- Probabilistic generative process: each label emits its token and produces the next label
- Model (emission + transition probabilities) learned from examples
- Typical extraction: most likely label sequence, given the tokens



Rule-Based Vs. Statistical

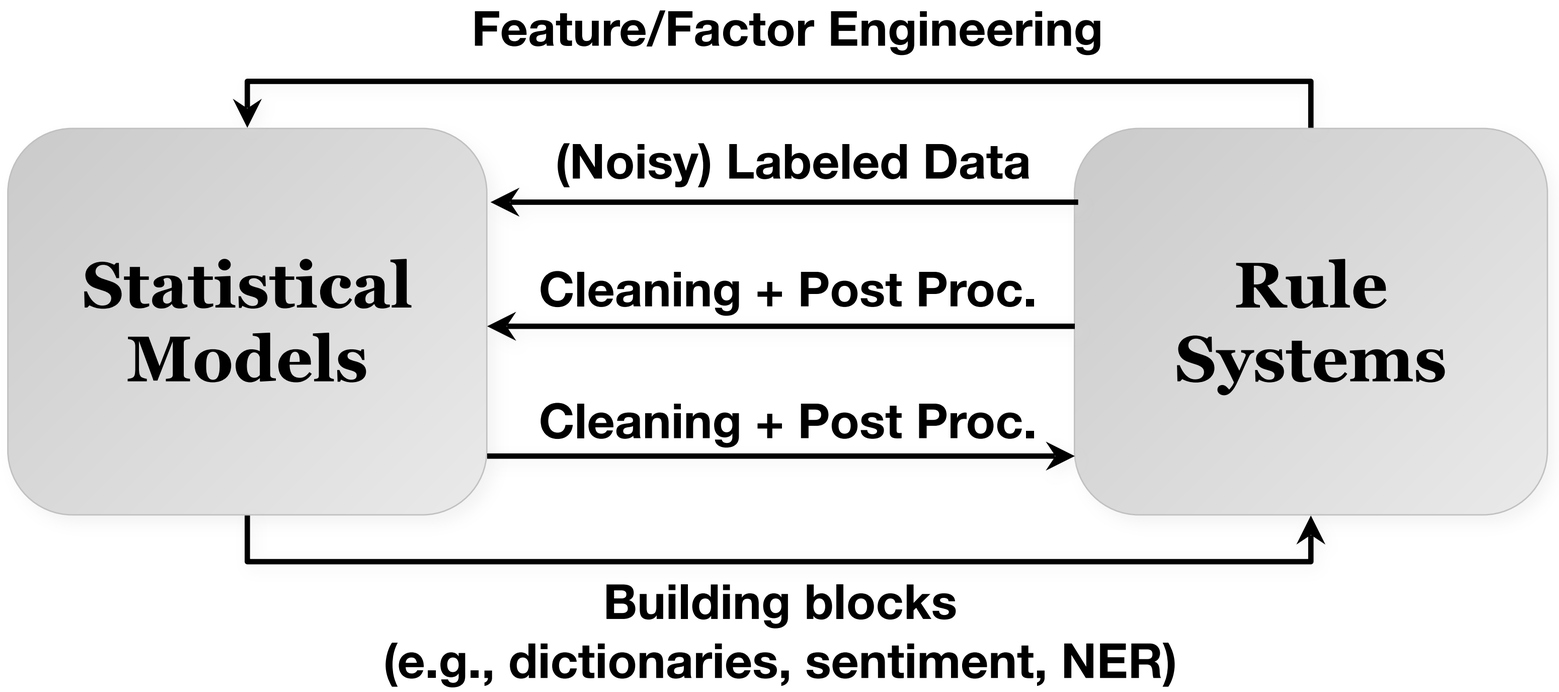
"[…] rules are effective, interpretable, and are easy to customize by non-experts to cope with errors."

Gupta & Manning, CONLL 2014



[Chiticariu, Li, Reiss, EMNLP'13]

Synergy between Rules and Statistics



The Document Spanners Formalism

What is Information Extraction?

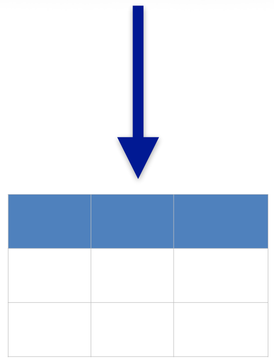
data-in-text (unstructured) → **data-in-db (structured)**

"Information Extraction (IE) is the name given to any process which selectively **structures and combines data** which is found, explicitly stated or implied, in one or more texts. The final output of the extraction process varies; in every case, however, it can be transformed so as to **populate some type of database**."

J. Cowie and Y. Wilks.
Handbook of Natural Language Processing 2000

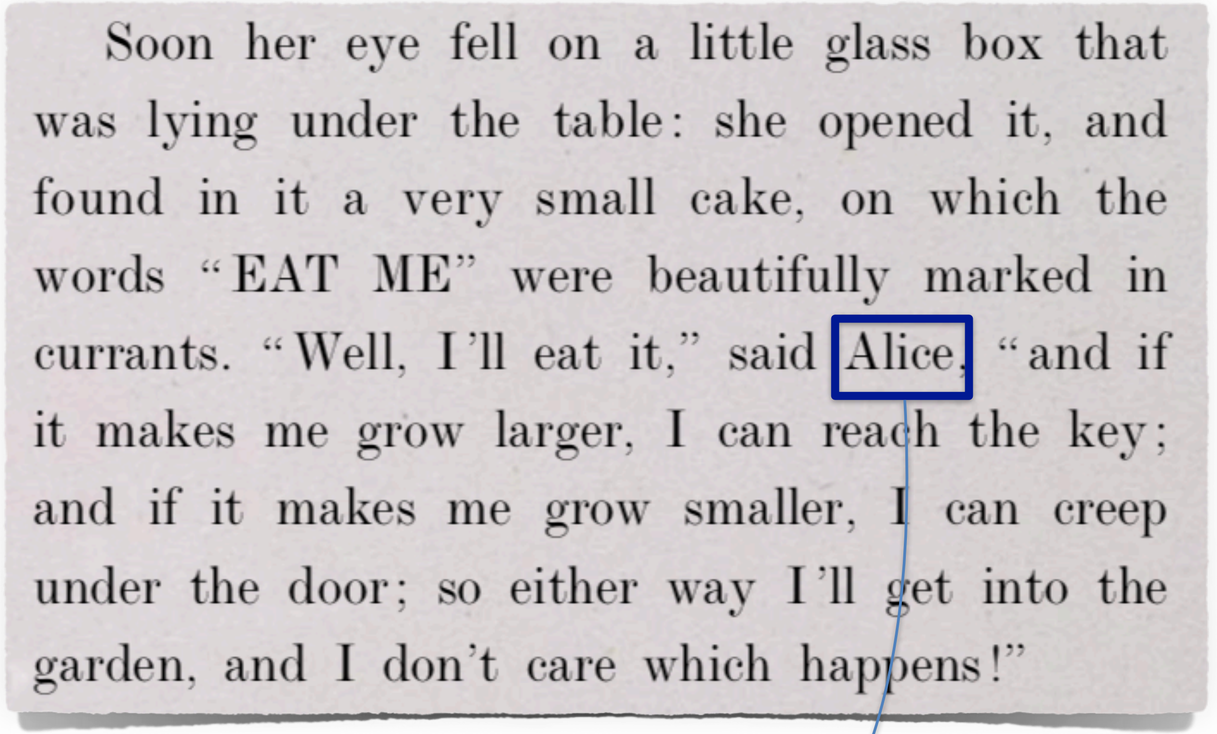
Information Extraction (IE)

Soon her eye fell on a little glass box that was lying under the table: she opened it, and found in it a very small cake, on which the words "EAT ME" were beautifully marked in currants. "Well, I'll eat it," said Alice, "and if it makes me grow larger, I can reach the key; and if it makes me grow smaller, I can creep under the door; so either way I'll get into the garden, and I don't care which happens!"



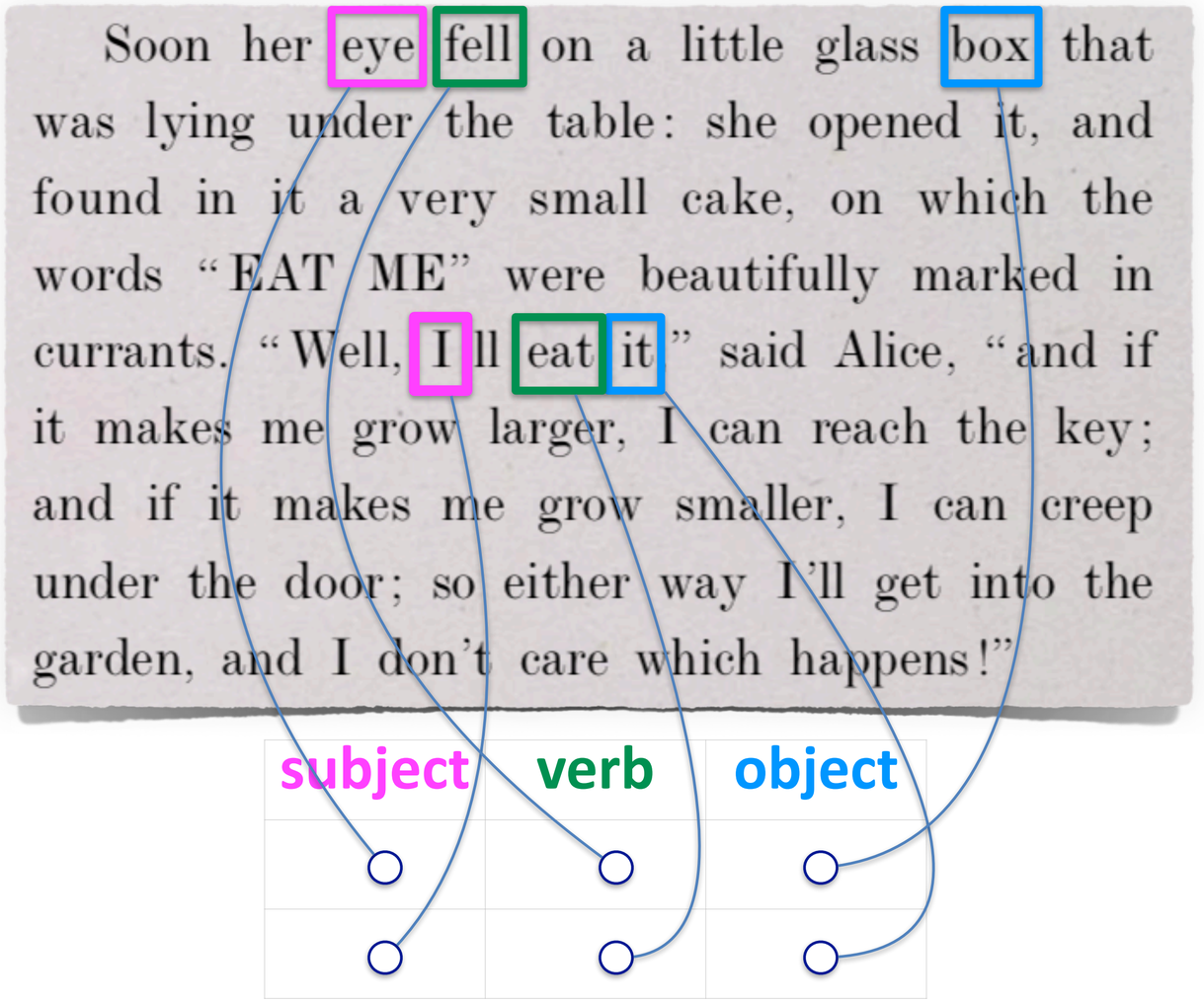
Information Extraction (IE)

Named Entity Recognition



Information Extraction (IE)

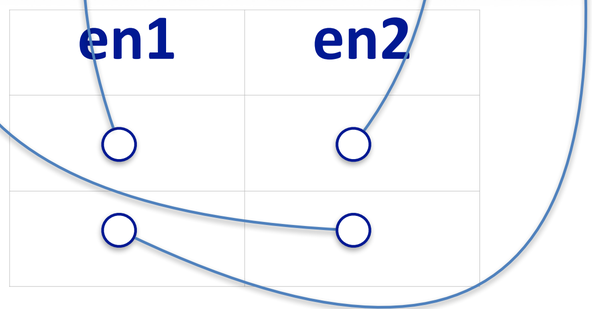
Relation Extraction



Information Extraction (IE)

Coreference Resolution

Soon her eye fell on a little glass box that was lying under the table: she opened it, and found in it a very small cake, on which the words "EAT ME" were beautifully marked in currants. "Well, I'll eat it," said Alice, "and if it makes me grow larger, I can reach the key; and if it makes me grow smaller, I can creep under the door; so either way I'll get into the garden, and I don't care which happens!"



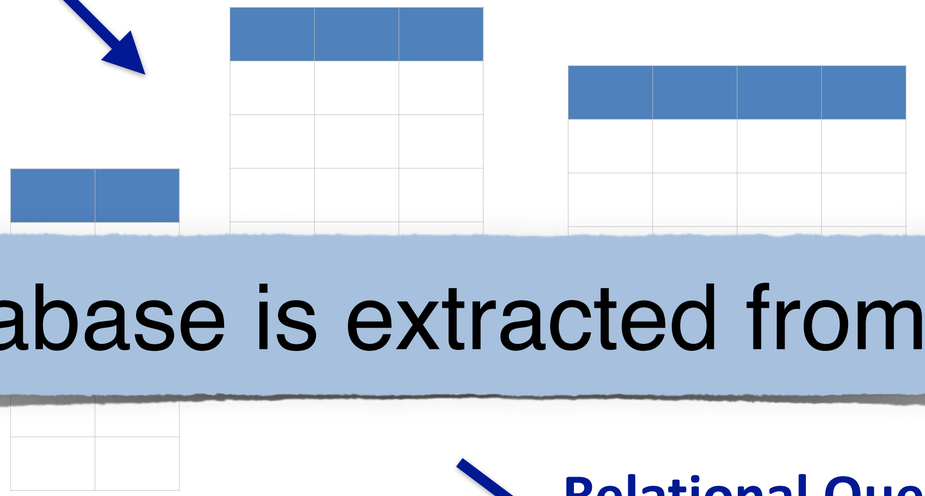
IE Queries

Soon her eye fell on a little glass box that was lying under the table: she opened it, and found in it a very small cake, on which the words "EAT ME" were beautifully marked in currants. "Well, I'll eat it," said Alice, "and if it makes me grow larger, I can reach the key; and if it makes me grow smaller, I can creep under the door; so either way I'll get into the garden, and I don't care which happens!"

Document

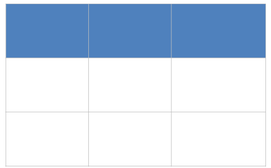
Extractors

Database



Database is extracted from text

Relational Query



Output

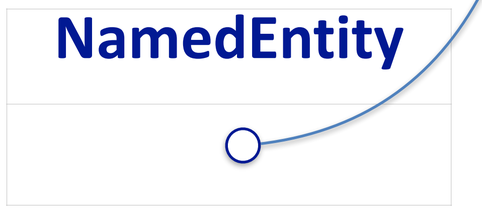
Will Alice eat the cake*?

*Assuming she always does what she says she will…

Information Extraction (IE)

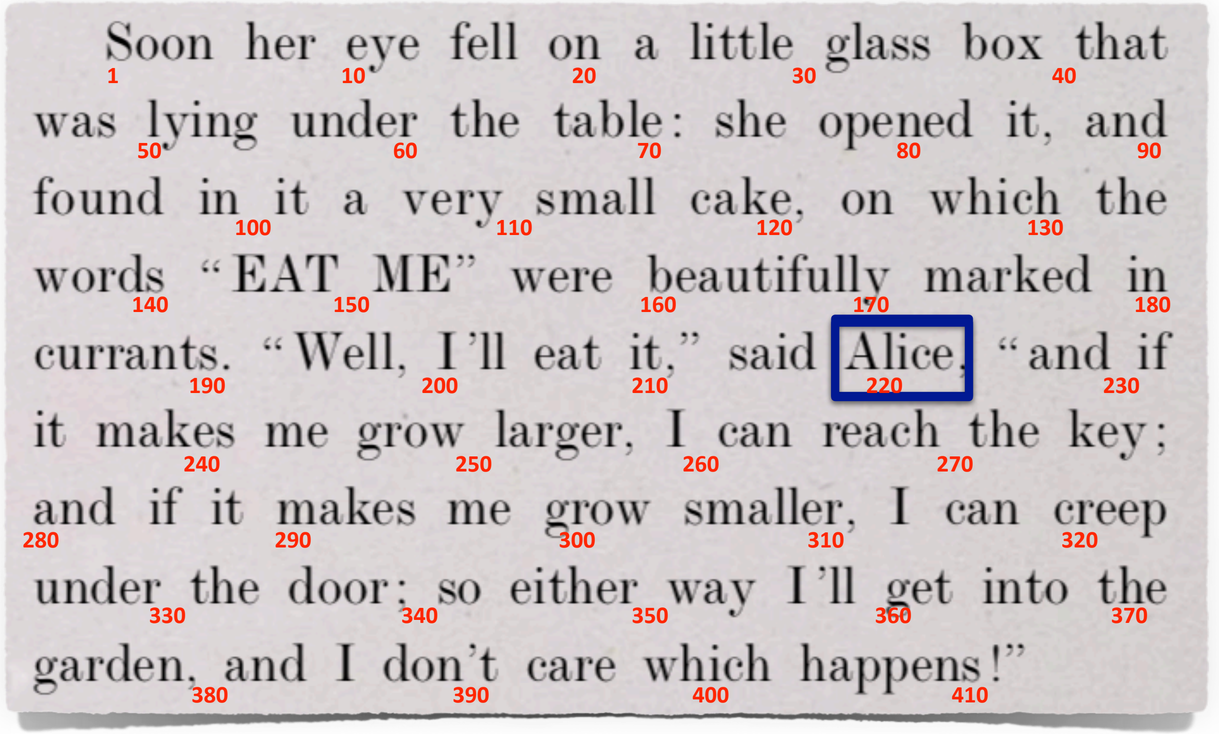
Named Entity Recognition

Soon her eye fell on a little glass box that was lying under the table: she opened it, and found in it a very small cake, on which the words "EAT ME" were beautifully marked in currants. "Well, I'll eat it," said Alice, "and if it makes me grow larger, I can reach the key; and if it makes me grow smaller, I can creep under the door; so either way I'll get into the garden, and I don't care which happens!"



Information Extraction (IE)

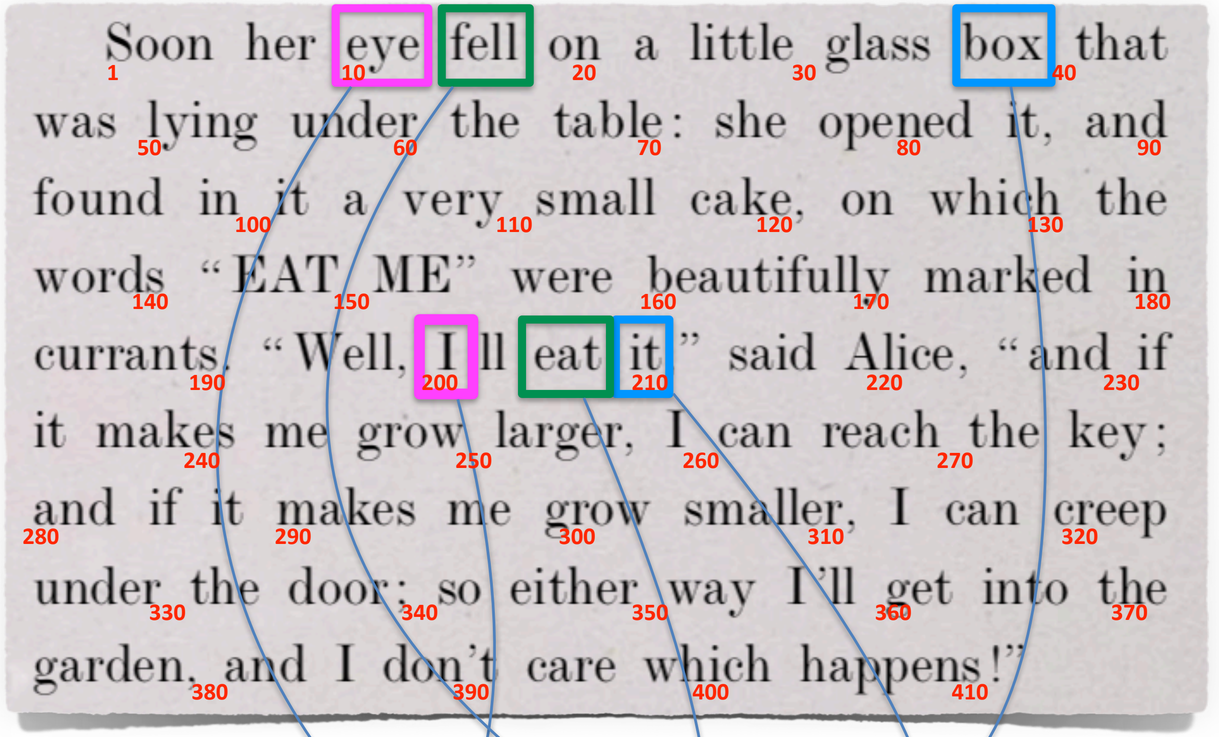
Named Entity Recognition



NamedEntity
[219,224)

Information Extraction (IE)

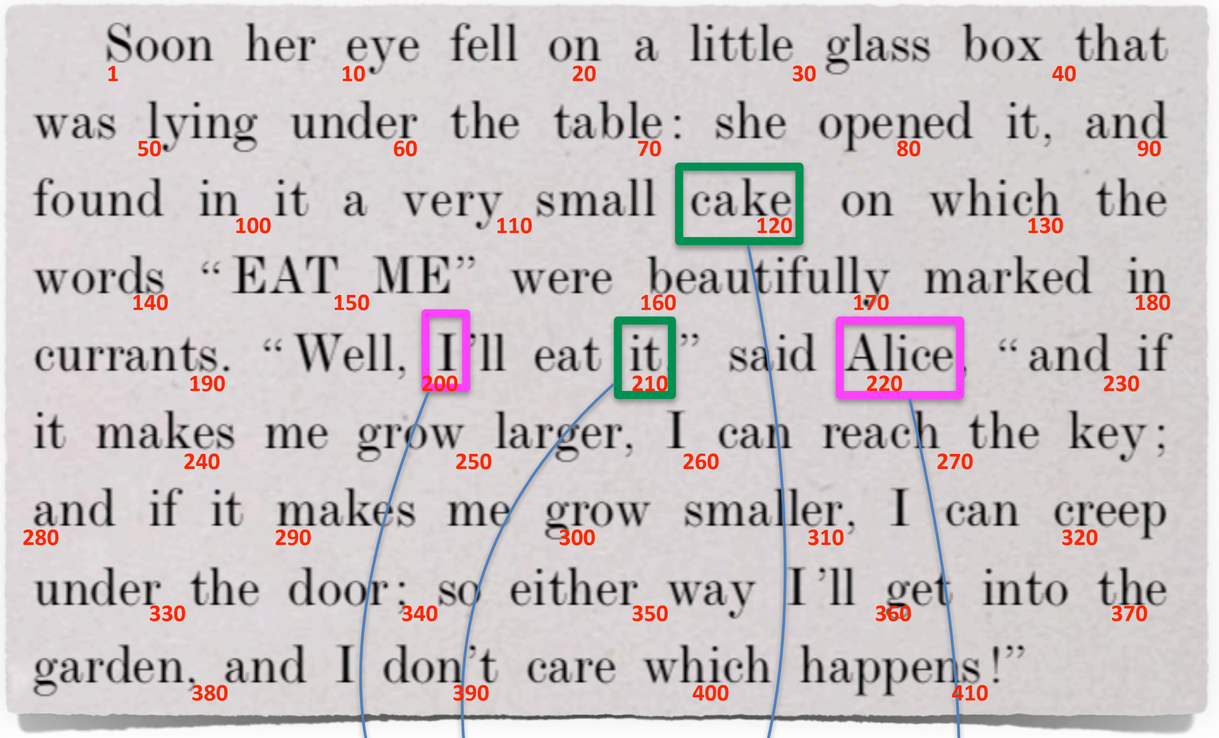
Relation Extraction



subject	verb	object
[10,13)	[14,18)	[37,40)
[200,201)	[205,208)	[209,211)

Information Extraction (IE)

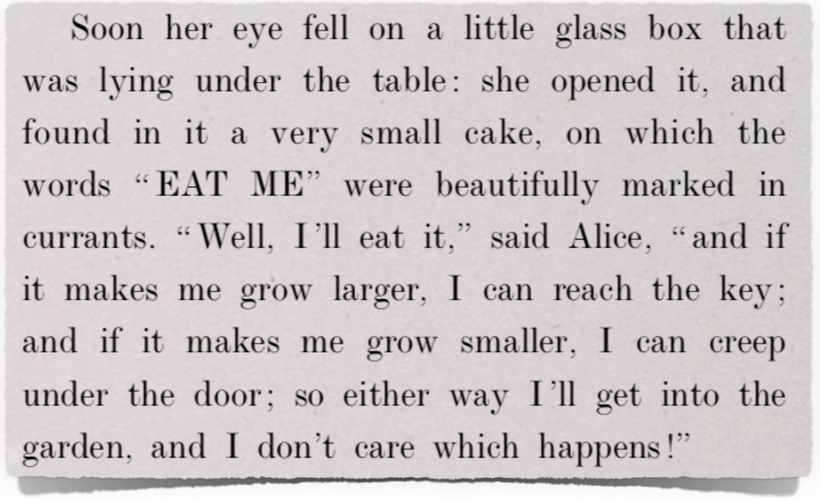
Coreference Resolution



en1	en2
[209,211)	[116,121)
[219,224)	[200,201)

IE Queries

Relational algebra over Extractors from text



Soon her eye fell on a little glass box that was lying under the table: she opened it, and found in it a very small cake, on which the words "EAT ME" were beautifully marked in currants. "Well, I'll eat it," said Alice, "and if it makes me grow larger, I can reach the key; and if it makes me grow smaller, I can creep under the door; so either way I'll get into the garden, and I don't care which happens!"

$$\text{NmEnt}(en) \bowtie \text{CorefRes}(en, subj) \bowtie \text{RelExt}(subj, \text{"eat"}, obj) \bowtie \text{CorefRes}(obj, \text{"cake"})$$

NmEnt

en
[219,224) Alice

CorefRes

en	subj
[209,211) it	[116,121) cake
[219,224) Alice	[200,201) I

RelExt

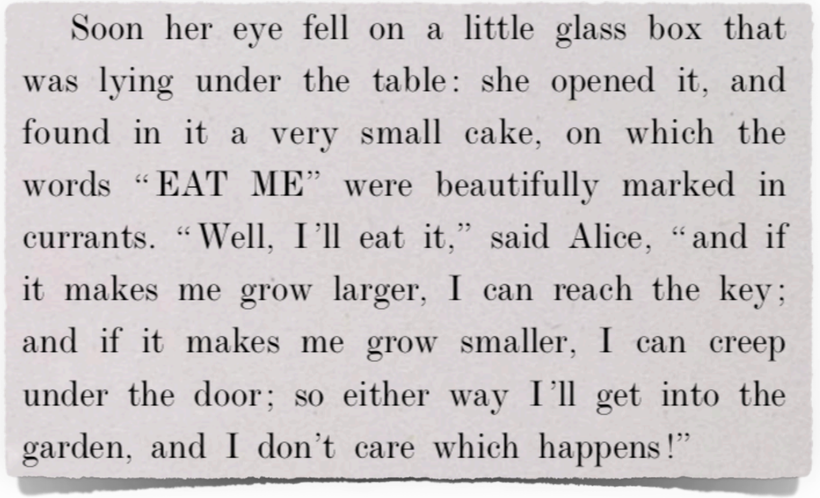
subj	verb	obj
[10,13) eye	[14,18) fell	[37,40) box
[200,201) I	[205,208) eat	[209,211) it

CorefRes

obj	obj'
[209,211) it	[116,121) cake
[219,224) Alice	[200,201) I

IE Queries

Relational algebra over Extractors from text



Soon her eye fell on a little glass box that was lying under the table: she opened it, and found in it a very small cake, on which the words "EAT ME" were beautifully marked in currants. "Well, I'll eat it," said Alice, "and if it makes me grow larger, I can reach the key; and if it makes me grow smaller, I can creep under the door; so either way I'll get into the garden, and I don't care which happens!"

$$\pi_{\varnothing}\Big(\text{NmEnt(en)} \bowtie \text{CorefRes(en, subj)} \bowtie \text{RelExt(subj, "eat", obj)} \bowtie \text{CorefRes(obj, "cake")}\Big)$$

NmEnt

en
[219,224) Alice

CorefRes

en	subj
[209,211) it	[116,121) cake
[219,224) Alice	[200,201) I

RelExt

subj	verb	obj
[10,13) eye	[14,18) fell	[37,40) box
[200,201) I	[205,208) eat	[209,211) it

CorefRes

obj	obj'
[209,211) it	[116,121) cake
[219,224) Alice	[200,201) I

en	subj	verb	obj	obj'
[219,224)	[200,201)	[205,208)	[209,211)	[116,121)

Yes!

IE Queries
in the Document Spanners framework

Soon her eye fell on a little glass box that was lying under the table: she opened it, and found in it a very small cake, on which the words "EAT ME" were beautifully marked in currants. "Well, I'll eat it," said Alice, "and if it makes me grow larger, I can reach the key; and if it makes me grow smaller, I can creep under the door; so either way I'll get into the garden, and I don't care which happens!"

Document

Extractors

Database

[3,13)	[13,15)	[3,13)	[2,4)
[32,39)	[3,13)	[42,48)	[3,13)

[54,89)	[13,15)

[42,48)	[2,4)	[3,13)

Database is extracted from text

Relational Query

[3,13)	[2,4)	[13,15)
[54,89)	[42,48)	[32,39)

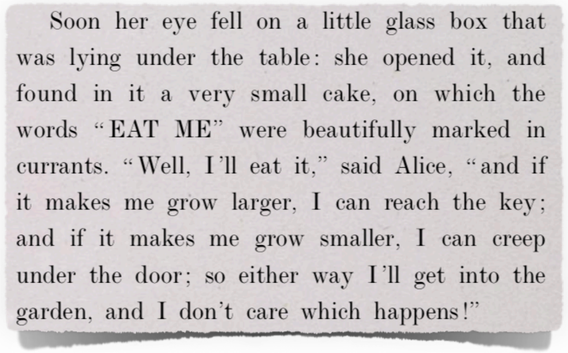
Output

$$\pi_x\left(\alpha_1(x,y) \bowtie \alpha_2(y,z) \bowtie \alpha_3(z)\right)$$

Document Spanners

Def: A **document spanner** is a function that maps every string into a relation over its spans

- Finite alphabet $\sum$ of symbols
- A spanner maps d $\in \sum^*$ into a relation over the spans [i,j) of d
  - [i,j) refers to the interval of d from symbol i (inclusive) to j (exclusive)
- The relation has a fixed signature (set of variables / attributes)



Soon her eye fell on a little glass box that was lying under the table: she opened it, and found in it a very small cake, on which the words "EAT ME" were beautifully marked in currants. "Well, I'll eat it," said Alice, "and if it makes me grow larger, I can reach the key; and if it makes me grow smaller, I can creep under the door; so either way I'll get into the garden, and I don't care which happens!"

Document d

[3,13)	[2,4)	[13,15)
[54,89)	[42,48)	[32,39)

Relation over the spans of d

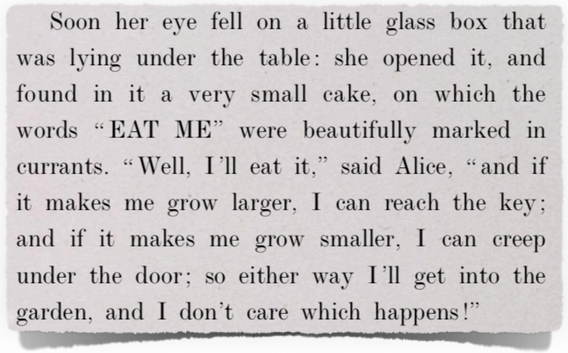
Instantiations

- The definition of a spanner is **abstract**
- Does not say what it extracts, how it is represented, and how it is executed
- Can be **generic NLP**: *tokenizer*, *POS tagger*, *sentence detector*, *dependency parser*, *NER*, …
  - e.g., NLTK, CoreNLP, OpenNLP, AllenNLP, …
- Can be **programmable**: dictionary, regex, automaton, deep network, SQL, etc.

Representation Systems for Document Spanners

Regular Spanners

Relational Algebra over Extension of Regular Expressions



Soon her eye fell on a little glass box that was lying under the table: she opened it, and found in it a very small cake, on which the words "EAT ME" were beautifully marked in currants. "Well, I'll eat it," said Alice, "and if it makes me grow larger, I can reach the key; and if it makes me grow smaller, I can creep under the door; so either way I'll get into the garden, and I don't care which happens!"

Document

Extension of Regular Expressions (regex formulas)

a*x{a*}a*

x
[3,13)
[42,48)
[54,89)

a*x{a*}a*y{a*}a*

x	y
[3,13)	[13,15)
[32,39)	[3,13)

Relational Algebra

[3,13)	[13,15)

Output

Regex Formulas

- Regex with embedded ("capture") variables

$$\gamma := \underbrace{\phi \mid \epsilon \mid \sigma \mid \gamma \vee \gamma \mid \gamma \bullet \gamma \mid \gamma^*}_{\text{Ordinary regex}} \mid \underbrace{x\{\gamma\}}_{\text{Span variable}}$$

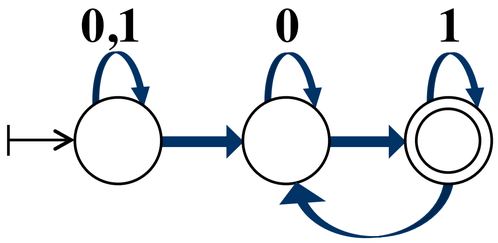
- Examples:
  - **.* in w{Alabama ∨ Alaska ∨ Arizona ∨ …} .***
  - **(.* z{[A-Z][a-z]*, y{[A-Z][a-z]*}} .*) | …**

- *Functionality* assumption: **each evaluation** (parse tree) **assigns one span to each and every variable**

  ⇒ Represents a spanner

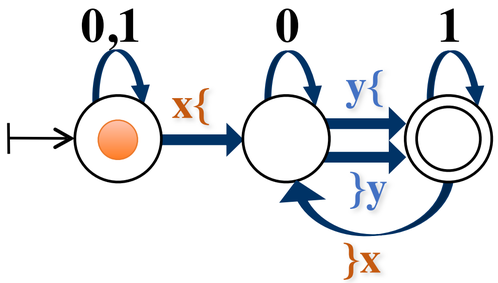
Variable-Set Automata

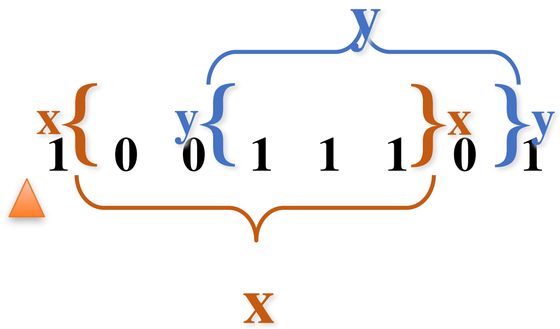
Ordinary NFA



1 0 0 1 1 1 0 1

Spanner Automaton





- ***Functionality*** **assumption: in an accepting run, each variable opens and later closes exactly once**
- **Nondeterministic ⇒ multiple accepting runs ⇒ multiple tuples**

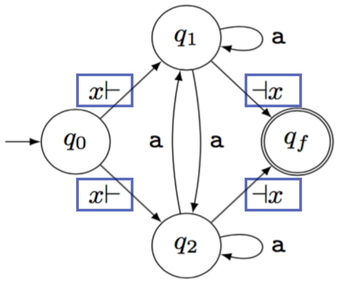
  ⇒ Represents a spanner

Regular Spanners

Soon her eye fell on a little glass box that was lying under the table: she opened it, and found in it a very small cake, on which the words "EAT ME" were beautifully marked in currants. "Well, I'll eat it," said Alice, "and if it makes me grow larger, I can reach the key; and if it makes me grow smaller, I can creep under the door; so either way I'll get into the garden, and I don't care which happens!"

Document

Finite set transducers (vset-automata)



x
[3,13)
[42,48)
[54,89)

[3,13)	[13,15)	[3,13)	[2,4)
[32,39)	[3,13)	[42,48)	[3,13)

[42,48)	[2,4)	[3,13)

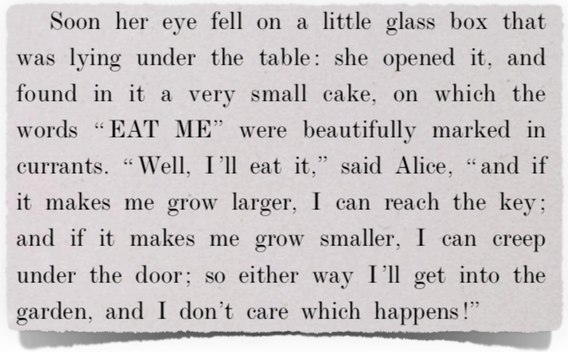
Relational Algebra

[3,13)	[2,4)	[13,15)
[54,89)	[42,48)	[32,39)

Output

Evaluation Approaches for Regular Spanners

IE Queries
in the Document Spanners framework


Soon her eye fell on a little glass box that was lying under the table: she opened it, and found in it a very small cake, on which the words "EAT ME" were beautifully marked in currants. "Well, I'll eat it," said Alice, "and if it makes me grow larger, I can reach the key; and if it makes me grow smaller, I can creep under the door; so either way I'll get into the garden, and I don't care which happens!"

Document

Extractors

Database

[3,13)	[13,15)	[3,13)	[2,4)
[32,39)	[3,13)	[42,48)	[3,13)

[54,89)	[13,15)

[42,48)	[2,4)	[3,13)

Database is extracted from text

Relational Query

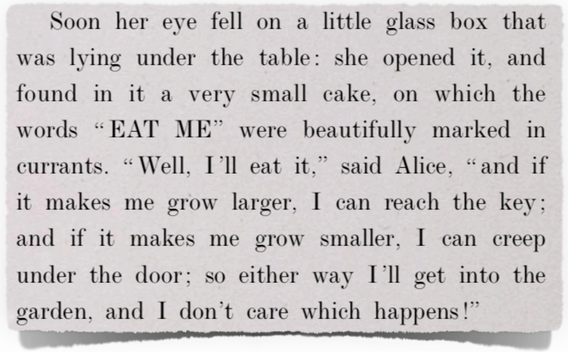
$$\pi_x\left(\alpha_1(x,y) \bowtie \alpha_2(y,z) \bowtie \alpha_3(z)\right)$$

[3,13)	[2,4)	[13,15)
[54,89)	[42,48)	[32,39)

Output

Naïve Approach

Materialize and evaluate


Soon her eye fell on a little glass box that was lying under the table: she opened it, and found in it a very small cake, on which the words "EAT ME" were beautifully marked in currants. "Well, I'll eat it," said Alice, "and if it makes me grow larger, I can reach the key; and if it makes me grow smaller, I can creep under the door; so either way I'll get into the garden, and I don't care which happens!"

Document

Extractors

Database

[3,13)	[32,39)
[42,48)	[2,4)
[54,89)	[13,15)

[3,13)	[13,15)	[3,13)	[2,4)
[32,39)	[3,13)	[42,48)	[3,13)

[42,48)	[2,4)	[3,13)

Materialize

Extract relations with basic extractors

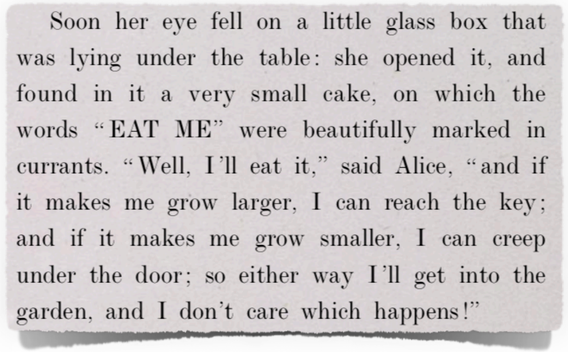
Relational Query

[3,13)	[2,4)	[13,15)
[54,89)	[42,48)	[32,39)

Output

Naïve Approach

Materialize and evaluate



Soon her eye fell on a little glass box that was lying under the table: she opened it, and found in it a very small cake, on which the words "EAT ME" were beautifully marked in currants. "Well, I'll eat it," said Alice, "and if it makes me grow larger, I can reach the key; and if it makes me grow smaller, I can creep under the door; so either way I'll get into the garden, and I don't care which happens!"

Document

Extractors

Database

[3,13)	[32,39)
[42,48)	[2,4)
[54,89)	[13,15)

[3,13)	[13,15)	[3,13)	[2,4)
[32,39)	[3,13)	[42,48)	[3,13)

[42,48)	[2,4)	[3,13)

Relational Query

Evaluate

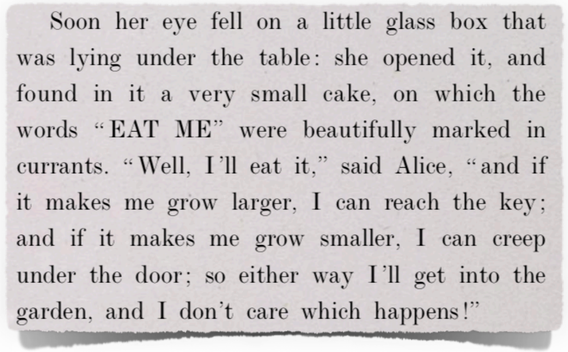
The relational query

[3,13)	[2,4)	[13,15)
[54,89)	[42,48)	[32,39)

Output

Naïve Approach

Materialize and evaluate


Soon her eye fell on a little glass box that was lying under the table: she opened it, and found in it a very small cake, on which the words "EAT ME" were beautifully marked in currants. "Well, I'll eat it," said Alice, "and if it makes me grow larger, I can reach the key; and if it makes me grow smaller, I can creep under the door; so either way I'll get into the garden, and I don't care which happens!"

Document

Extractors

Database

[3,13)	[32,39)
[42,48)	[2,4)
[54,89)	[13,15)

[3,13)	[13,15)	[3,13)	[2,4)
[32,39)	[3,13)	[42,48)	[3,13)

[42,48)	[2,4)	[3,13)

Relational Query

Materialize

Extract relations with basic extractors

Evaluate

The relational query

[3,13)	[2,4)	[13,15)
[54,89)	[42,48)	[32,39)

Output

! The size of intermediate database can exponential

! Evaluation of conjunctive queries is NP-hard

IE Queries

Relational algebra over Extractions from text

Do we really need to materialize the intermediate database?

Compilation Approach

Compile and run

Soon her eye fell on a little glass box that was lying under the table: she opened it, and found in it a very small cake, on which the words "EAT ME" were beautifully marked in currants. "Well, I'll eat it," said Alice, "and if it makes me grow larger, I can reach the key; and if it makes me grow smaller, I can creep under the door; so either way I'll get into the garden, and I don't care which happens!"

Document

Extractors

Database

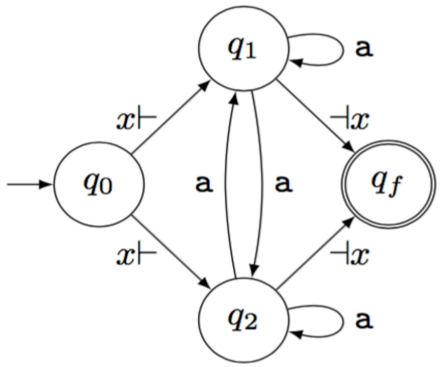
[3,13)	[32,39)
[42,48)	[2,4)
[54,89)	[13,15)

[3,13)	[13,15)	[3,13)	[2,4)
[32,39)	[3,13)	[42,48)	[3,13)

[42,48)	[2,4)	[3,13)

Compile

Into an automaton



Relational Query

[3,13)	[2,4)	[13,15)
[54,89)	[42,48)	[32,39)

Output

Compilation Approach

Compile and run

Soon her eye fell on a little glass box that was lying under the table: she opened it, and found in it a very small cake, on which the words "EAT ME" were beautifully marked in currants. "Well, I'll eat it," said Alice, "and if it makes me grow larger, I can reach the key; and if it makes me grow smaller, I can creep under the door; so either way I'll get into the garden, and I don't care which happens!"

Document

Extractors

Database

[3,13)	[32,39)
[42,48)	[2,4)
[54,89)	[13,15)

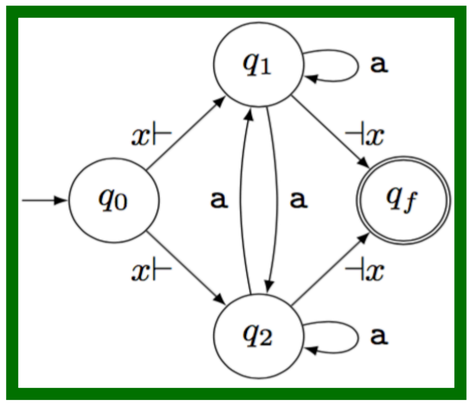
[3,13)	[13,15)	[3,13)	[2,4)
[32,39)	[3,13)	[42,48)	[3,13)

[42,48)	[2,4)	[3,13)

Relational Query

Run

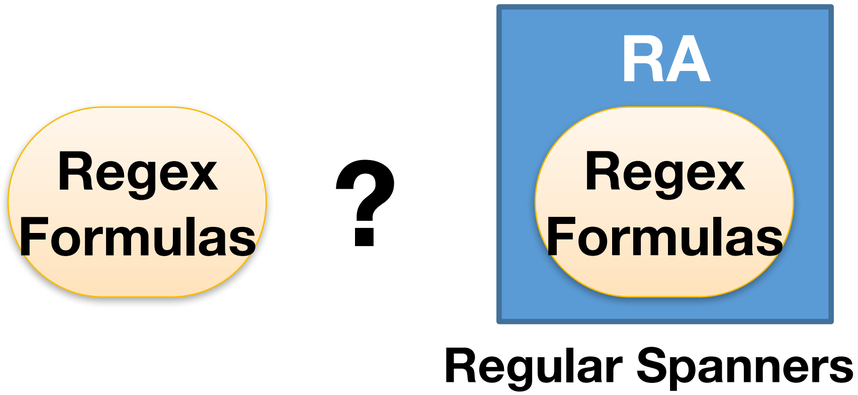
On the input document



[3,13)	[2,4)	[13,15)
[54,89)	[42,48)	[32,39)

Output

Classes of Spanners

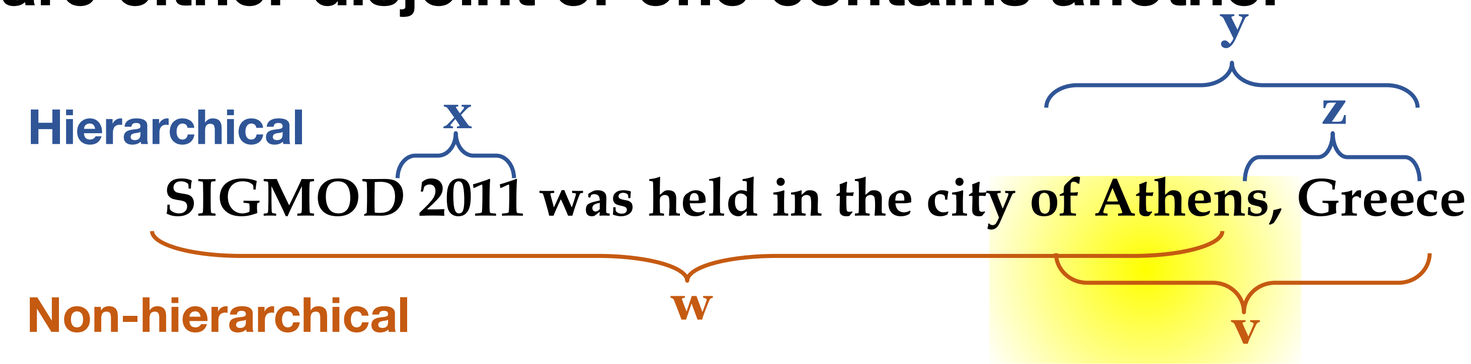


What expressive power does the Relational Algebra add to the class of Regex formulas?

Hierarchical Spanners

Def. A spanner is **hierarchical** if its tuples are "balanced" (like parentheses) for all input documents.

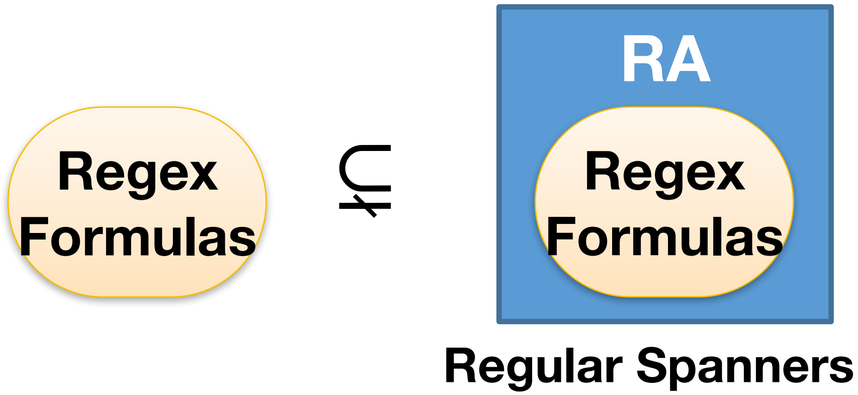
∀ docs d, tuples $t \in P(d)$, vars x,y, spans $t(x)$ and $t(y)$ are either disjoint or one contains another



OBS. Regex formulas can express only hierarchical spanners; regular spanners are not necessarily hierarchical.

THM. Regex formulas can express precisely the hierarchical regular spanners.

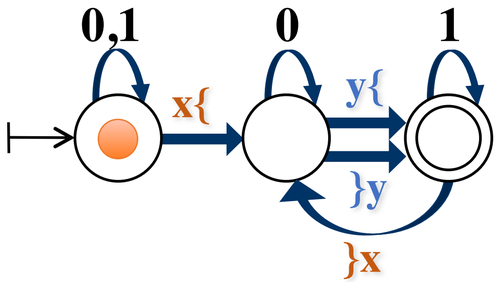
[Fagin, K, Reiss, Vansummeren 2015]



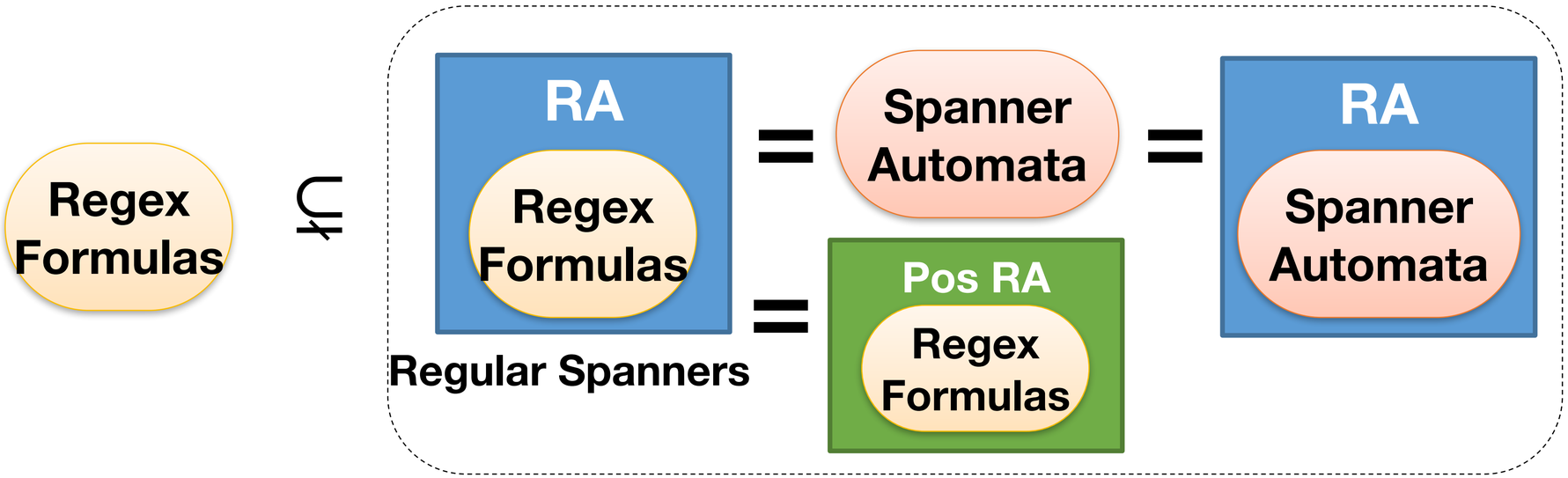
Regular Spanners

Regular Spanner Representations

THM. A spanner is regular iff it is expressible as a spanner automaton.

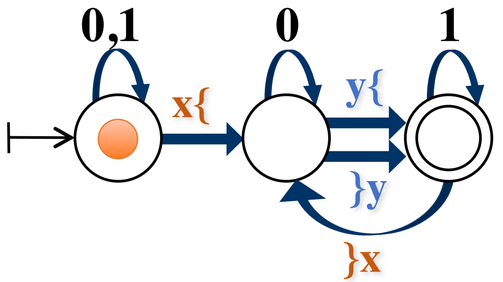


[Fagin, K, Reiss, Vansummeren 2015]


Regex Formulas
⊆
RA
Regex Formulas
=
Spanner Automata
=
RA
Spanner Automata
Regular Spanners
=
Pos RA
Regex Formulas

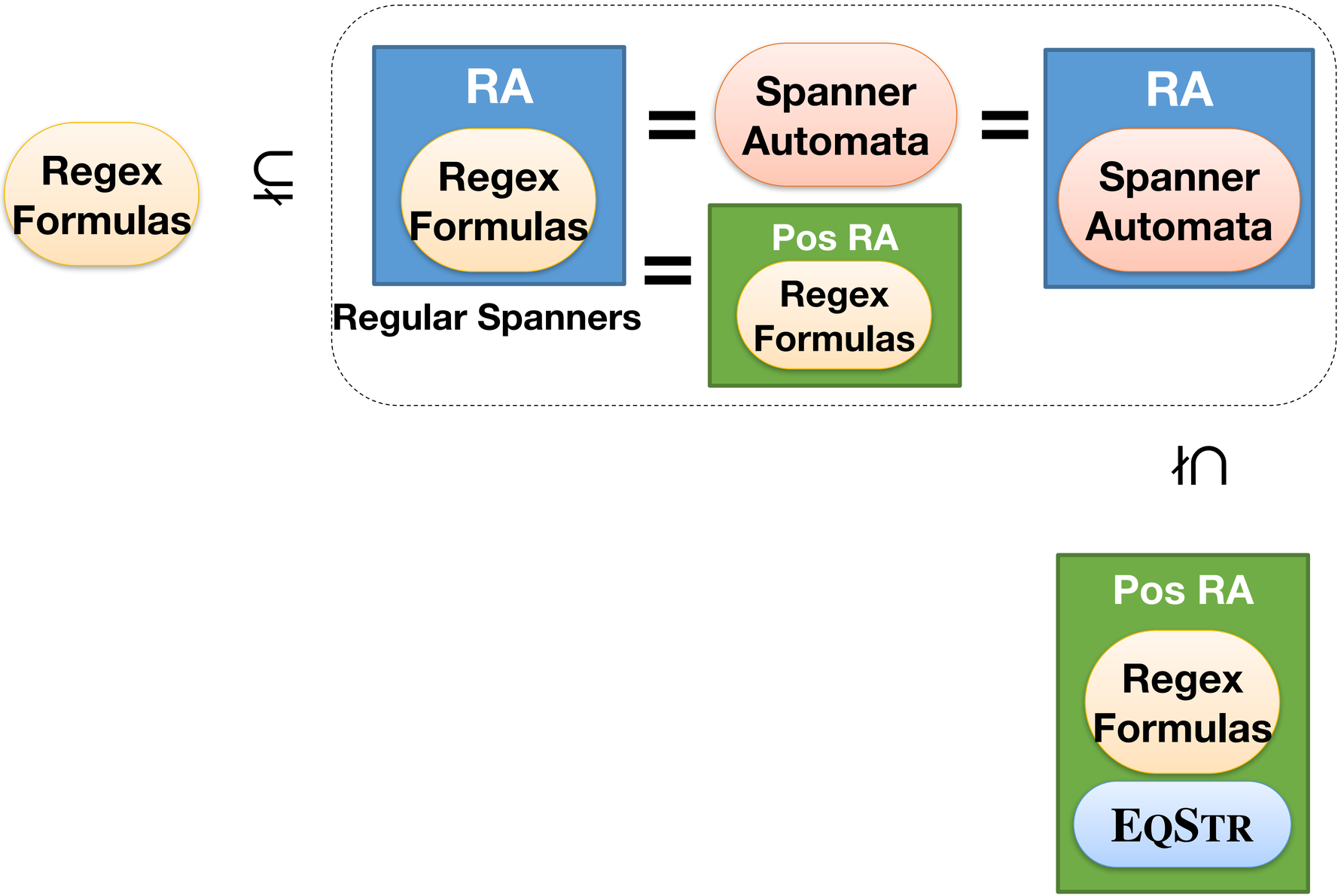
Regular Spanner Representations

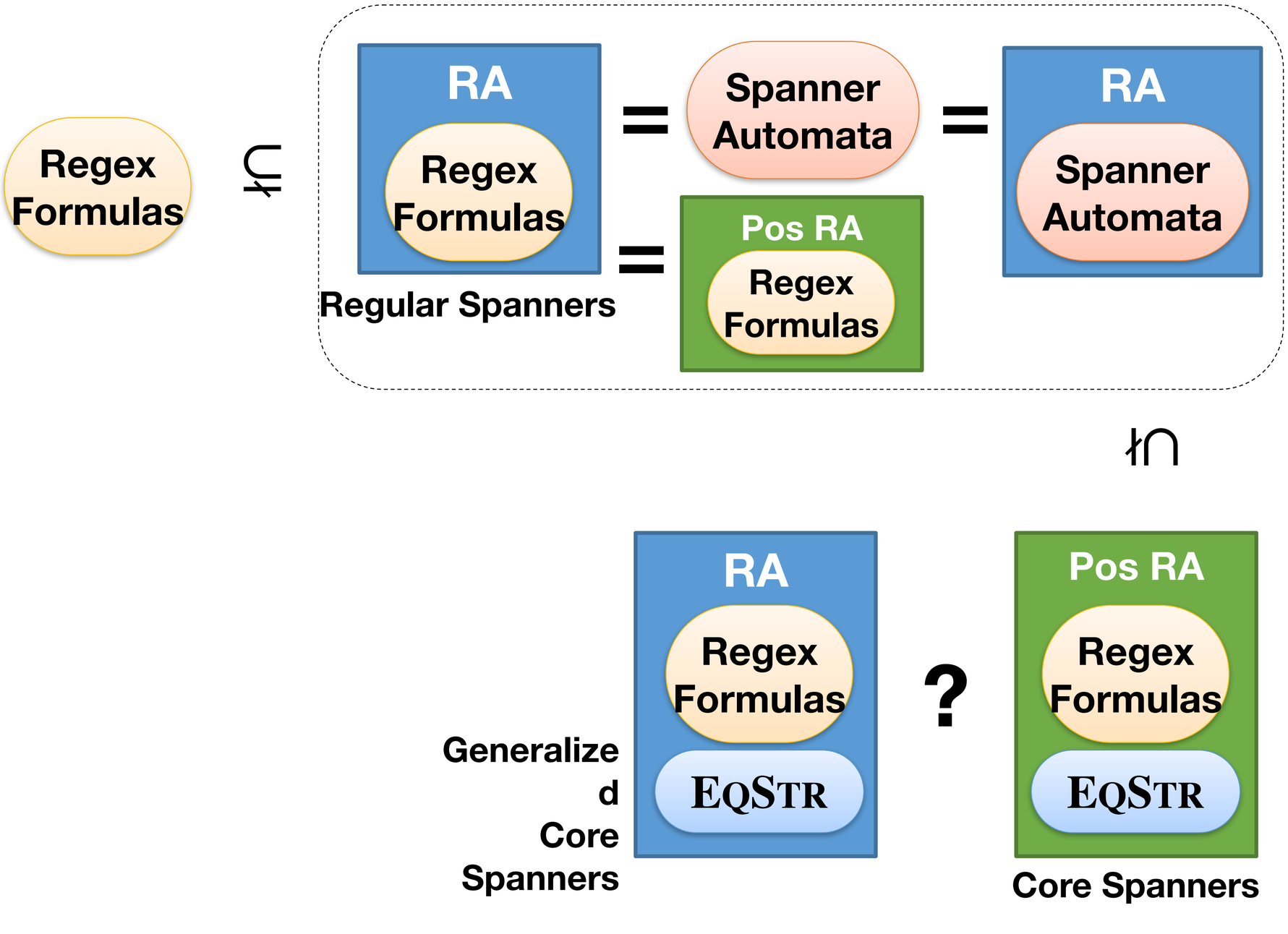
THM. **A spanner is regular iff it is expressible as a spanner automaton.**



COR. **EQSTR is not a regular spanner.**

[Fagin, K, Reiss, Vansummeren 2015]


Regex Formulas
⊂
RA
Regex Formulas
=
Spanner Automata
=
RA
Spanner Automata
=
Pos RA
Regex Formulas
Regular Spanners
⊂
Pos RA
Regex Formulas
EQSTR
Core Spanners


Regex Formulas
⊆
RA
Regex Formulas
=
Spanner Automata
=
RA
Spanner Automata
=
Pos RA
Regex Formulas
Regular Spanners
⊆
⊆
Generalized Core Spanners
RA
Regex Formulas
EQSTR
?
Pos RA
Regex Formulas
EQSTR
Core Spanners

Difference in Core Spanners

- Are core spanners closed under difference?
- Indication of "not": Only **positive** RA is used
- Indication of "yes": Positive RA over Regex (without EQSTR) is closed under difference
- Candidate for non-closure proof: string inequality

THM. String inequality is a core spanner.

?

[Fagin, K, Reiss, Vansummeren 2015]

The Answer

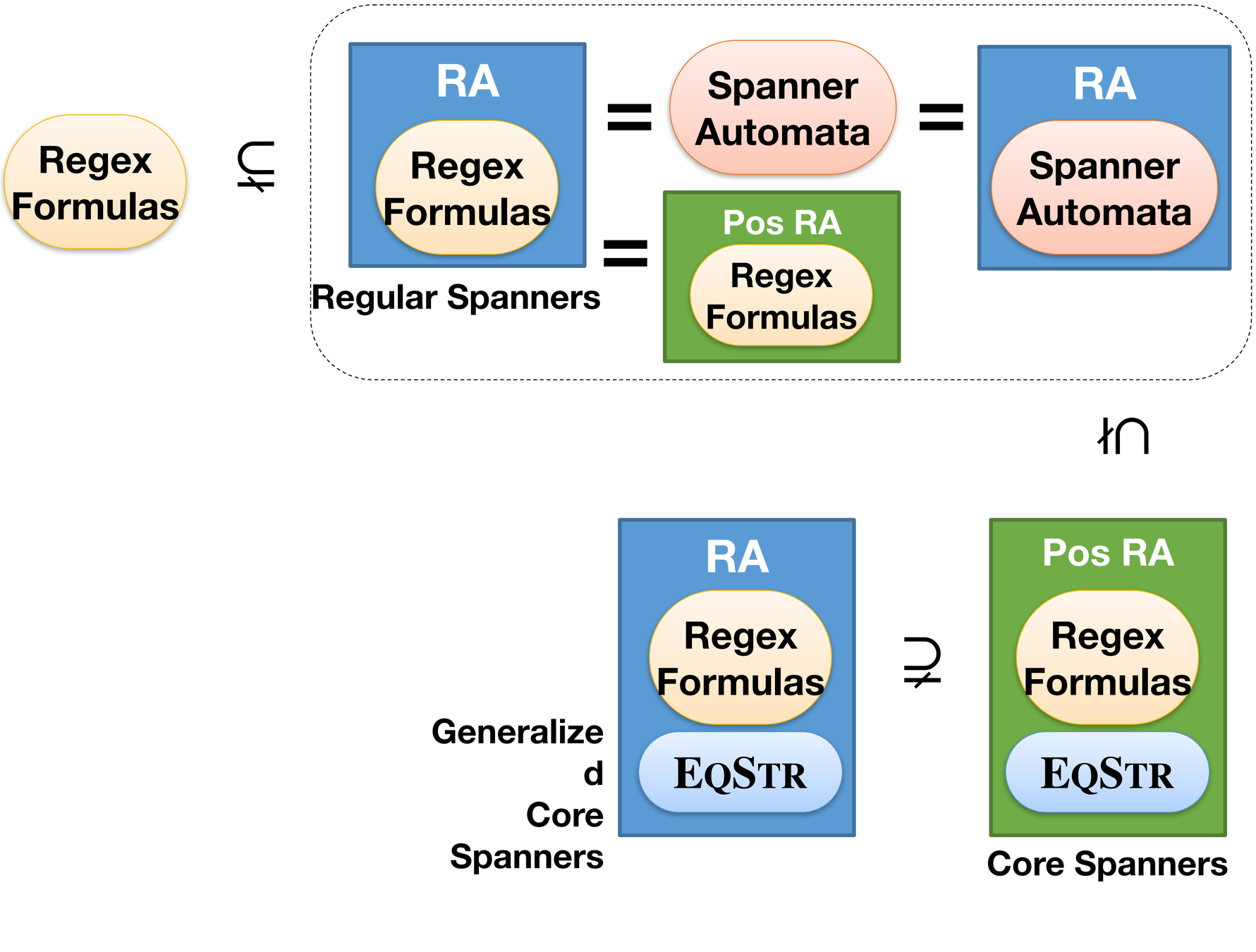
THM. **The substring relation is expressible as a core spanner.**

Find all spans x and y such that $\mathrm{str}(x)$ is a substring of $\mathrm{str}(y)$

THM. **The non-substring relation is not expressible as a core spanner.**

COR. **The class of core spanners is not closed under difference.**

[Fagin, K, Reiss, Vansummeren 2015]


Regex Formulas
RA
Regex Formulas
Spanner Automata
RA
Spanner Automata
Pos RA
Regex Formulas
Regular Spanners
Generalized Core Spanners
RA
Regex Formulas
EQSTR
Pos RA
Regex Formulas
EQSTR
Core Spanners

There are other classes of spanners

- Incorporating recursion - Datalog over regex formulas - captures polynomial spanners-
  - Every spanner that can be evaluated in polynomial time can be defined as such program
  - And vice-versa
- Spanners based on context free grammars
- And more…

Computational Complexity

Classic Database Complexity Measures

Data Complexity

Input: text

- Spanner fixed

Problem: regex formulas can be quite large…

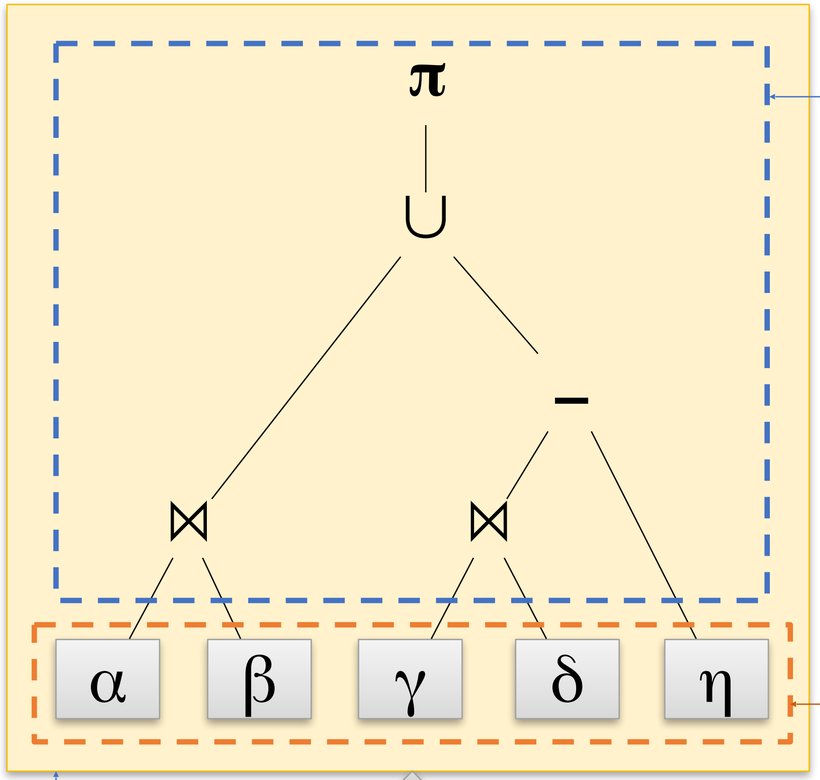
Combined Complexity

Input: text, spanner

- Nothing fixed

Problem: hardness already in the rel. model

The Computational Problem



Relational query

x	y	z
[1,14⟩	[30,36⟩	[1,36⟩
[42,47⟩	[52,65⟩	[42,65⟩
[102,110⟩	[115,125⟩	[102,125⟩

Output

Atomic extractors
(e.g., regex formulas)

Spanner representation

text

Input document

Regex Examples (RFC 2822)

Date format

```
^(?:\s*(Sun|Mon|Tue|Wed|Thu|Fri|Sat),\s*)?(0?[1-9]|[1-2][0-9]|3[01
])\s+(Jan|Feb|Mar|Apr|May|Jun|Jul|Aug|Sep|Oct|Nov|Dec)\s+(19[0-9]{
2}|[2-9][0-9]{3}|[0-9]{2})\s+(2[0-3]|[0-1][0-9]):([0-5][0-9])(?::(
60|[0-5][0-9]))?\s+([-\+][0-9]{2}[0-5][0-9]|(?:UT|GMT|(?:E|C|M|P)(
?:ST|DT)|[A-IK-Z]))(\s*\((\\(|\\)|(?<=[^\\])\((?<C>)|(?<=[^\\])\
)(?<-C>)|[^\(\)]*)*(?(C)(?!))\))*\s*$
```

Mailbox format

```
^((?>[a-zA-Z\d!#$%&'*+\-/=?^_`{|}~]+\x20*|"((?=[\x01-\x7f])[^"\\]|
\\[\x01-\x7f])*"\x20*)*(?<angle><))?((?!\.)(?>\.?[a-A-Z\d!#$%&’*+\
-/=?^_`{|}~]+)+|"((?=[\x01-\x7f])[^"\\]|\\          [\x01          -
\x7f])*")@(((?!-)[a-zA-Z\d\-]+(?<!-)\.)+[a-zA-Z]  {2,}|\[(((  ?(?<!\
[)\.)(25[0-5]|2[    0-4]\d|[01]?\d?\d)){4}|[a-zA-Z\d\-]*[a-zA-Z\d]:
((?=[\x01-\x7f])[^\ \\[\]]|\\[\x01-\x7f])+)\])(?(angle)>)$
```

New Measure: Extraction Complexity

Data Complexity

Input: text

- Spanner fixed

Problem: regex formulas can be quite large…

Combined Complexity

Input: text, spanner

- Nothing fixed

Problem: hardness already at the rel. model

Extraction Complexity

Input: text, atomic spanners

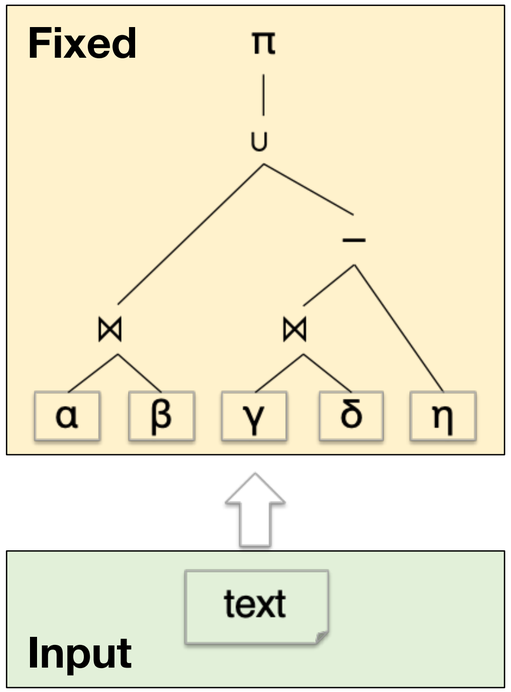
- Relational query fixed

Complexity Measures

Data Complexity

Input: text

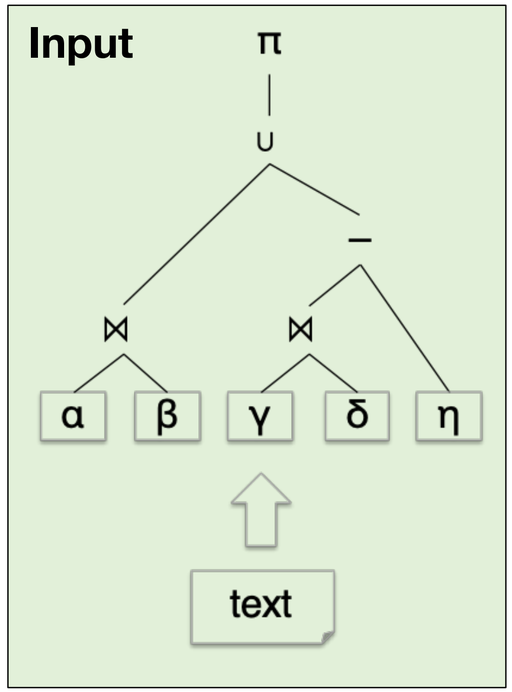
- Spanner fixed



Combined Complexity

Input: text, spanner

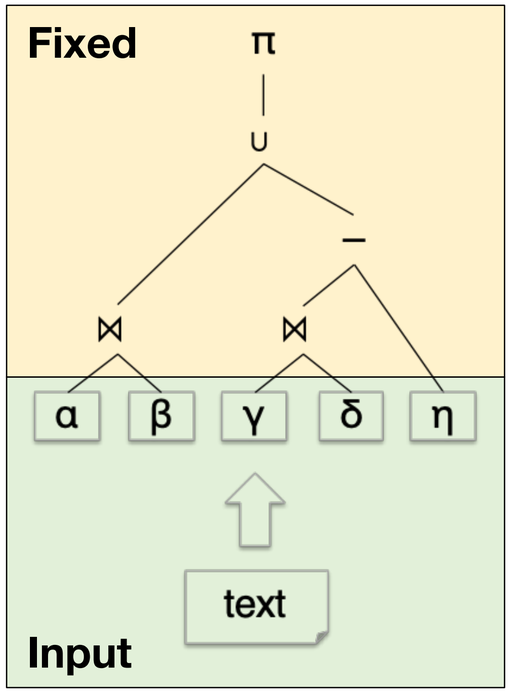
- Nothing fixed



Extraction Complexity

Input: text, atomic spanners

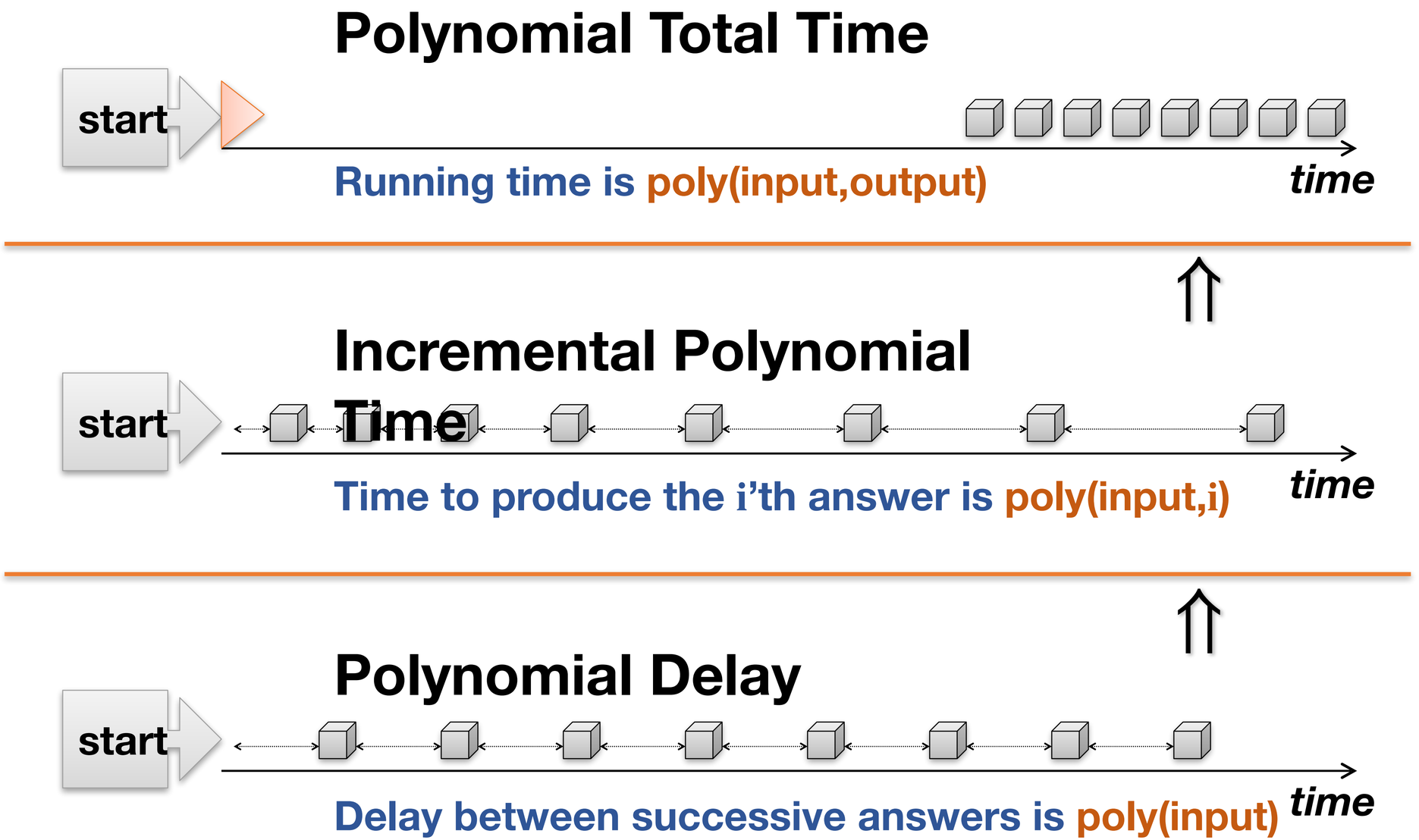
- Relational query fixed



What happens when the output is too big

- In combined/extraction complexity, the size of the output can be exponential in that of the input
- More precisely, the spanner $\alpha[x_1,\ldots,x_m]$ can have $|d|^{O(m)}$ answers on a document d
- Hence, “polynomial time” is not a proper yardstick of tractability
- We need an **output-sensitive** measure that accounts for both the input and output size

Background: Tractability of Enumeration



Complexity of Atomic Regular Spanners

The following apply to regex formulas / spanner automata:

Thm. [comb./ext. complexity] Answers can be enumerated with polynomial delay.

[Freydenberger, K, Peterfreund 2018]

Thm. [data complexity] Answers can be enumerated with constant delay after a linear preprocessing phase.

(but exponential in the automaton/regex size)

[Florenzano, Riveros, Ugarte, Vansummeren, Vrgoc: *Constant Delay Algorithms for Regular Document Spanners*. PODS 2018]

Thm. Answers can be enumerated with constant delay after a linear preprocessing phase, where *all times are polynomial in the size of the spanner.*

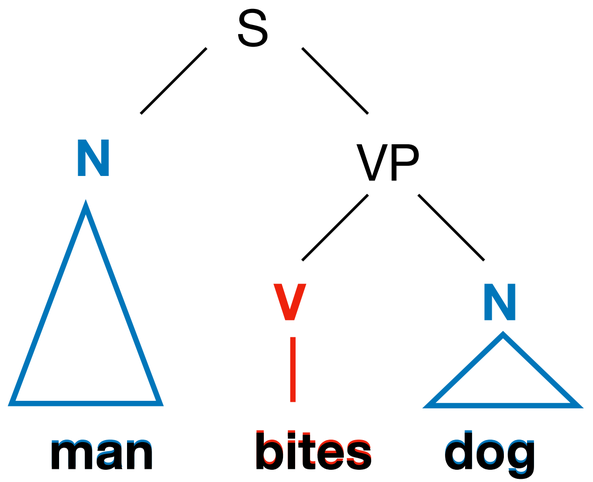
[Amarilli, Bourhis, Mengel, Niewerth: *Constant-Delay Enumeration for Nondeterministic Document Spanners.* ICDT 2019]

Context Free Spanners

Grammar-Based Document Spanners

Context free grammars (CFGs) use rules to parse languages

(1) S ⟶ N VP
(2) VP ⟶ V N
(3) N ⟶ **man** | **dog**
(4) V ⟶ **bites**



Definition

Extraction Grammars:

CFGs with **variable markers x{ , }x**…

S ⟶ **x{**N**}x y{**VP**}y**

x	y
man	bites dog

x{man}x y{bites dog}y

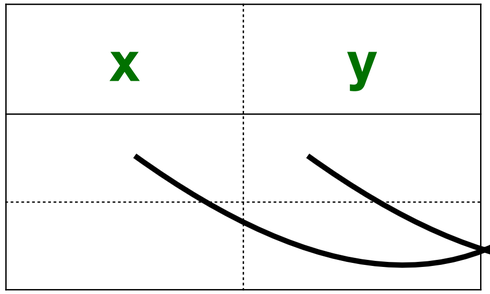
Extraction Grammars

(1) S → B x{ A }y B
(2) A → aAa
(3) A → }x {y
(4) B → epsilon | aB

terminals: a , x{ , x} , y{ , y}
non-terminals: S , A , B

aaa

spanner



~~x{a}x y{a}y~~
x{a}x y{a}y a
a x{a}x y{a}y

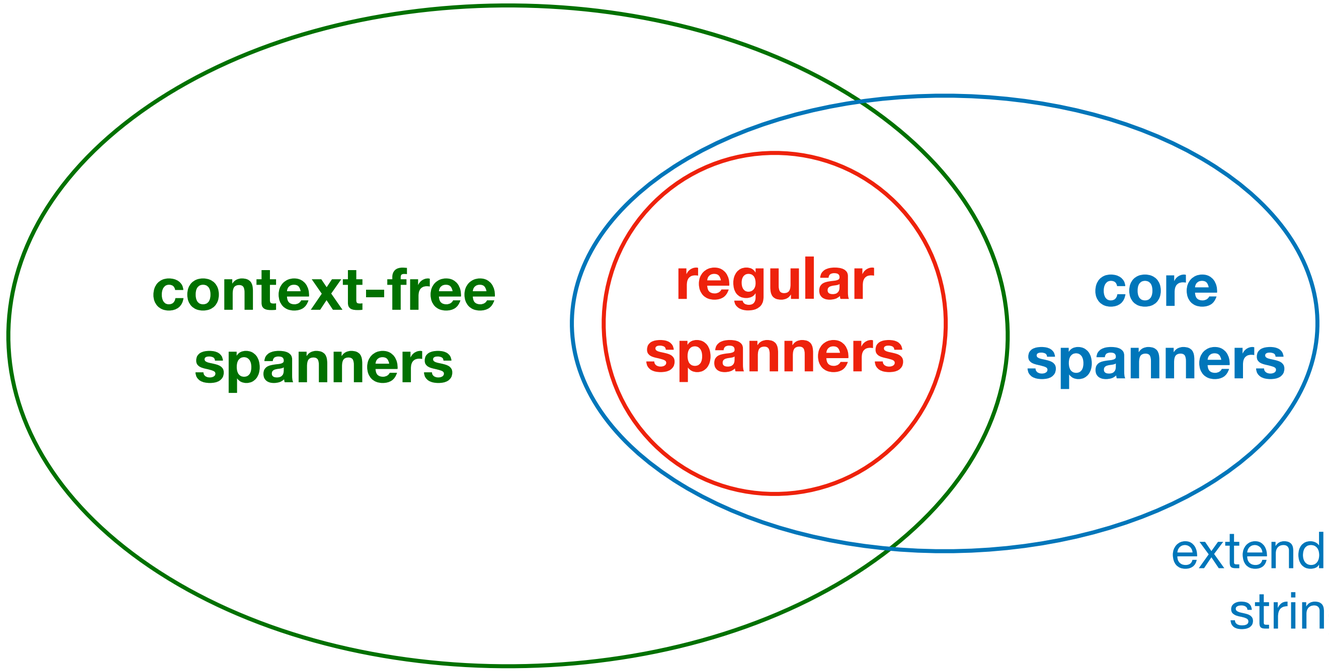
Extracts (x,y) whose corresponding substrings are adjacent and of the same length.

Context-Free Spanners - Expressiveness

Definition

Spanners expressible by **extraction grammars**
(or **pushdown extraction automata**)

Proposition



extend regular with
string equality

expressible by
finite state automata

Context-Free Spanners - Evaluation Complexity

Theorem

For every extraction grammar G with k variables and document d, one can output extracted relation in

$$O(|d|^{2k+3} k^3 |G| + |G|^2)$$

The exponent depends on the number of variables in the extracted relation

Can we decrease the exponent?

Context-Free Spanners - Enumeration of Extractions

Theorem

For every unambiguous extraction grammar G with k variables and document d there is an algorithm that outputs the tuples of the extracted relation with

- preprocessing $O(|d|^5 |G|^2 3^{4k})$
- delay O(k)

each output tuple is induced only once

- Data complexity:
  - polynomial preprocessing and constant delay
  - The delay is independent of the document

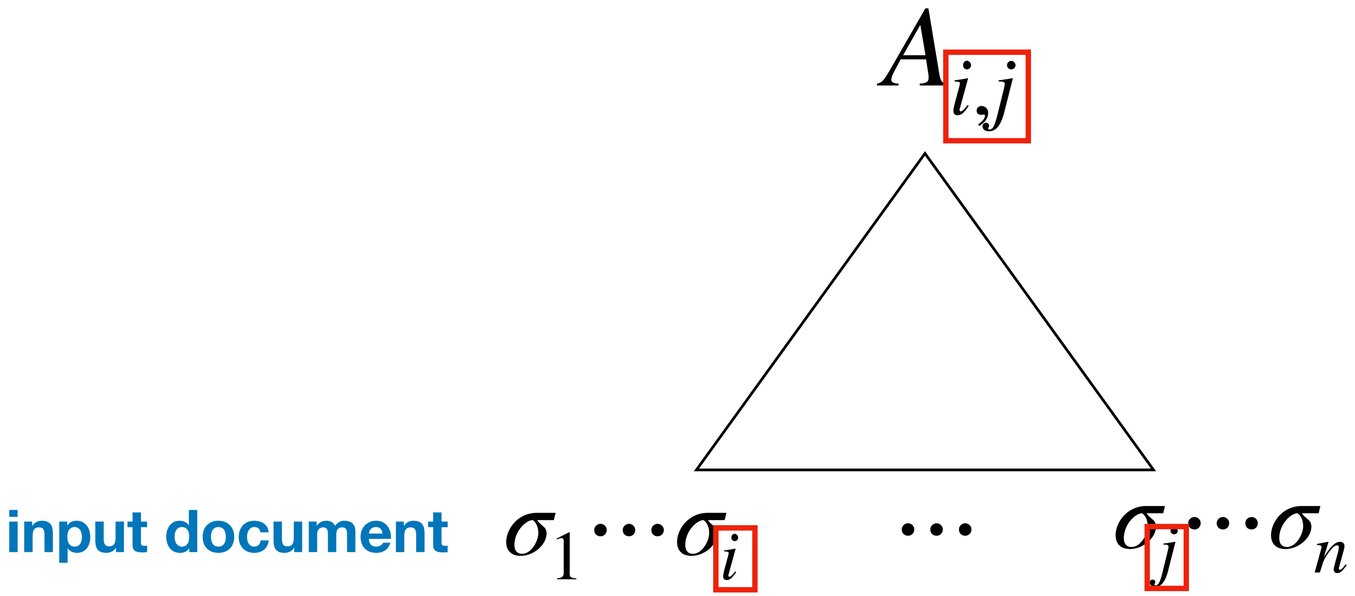
Note that for regular spanners preprocessing is linear*

*[Amarilli, Bourhis, Mengel, Niewerth, ICDT19]

Enumeration Algorithm - Preprocessing 1

adjustment to document

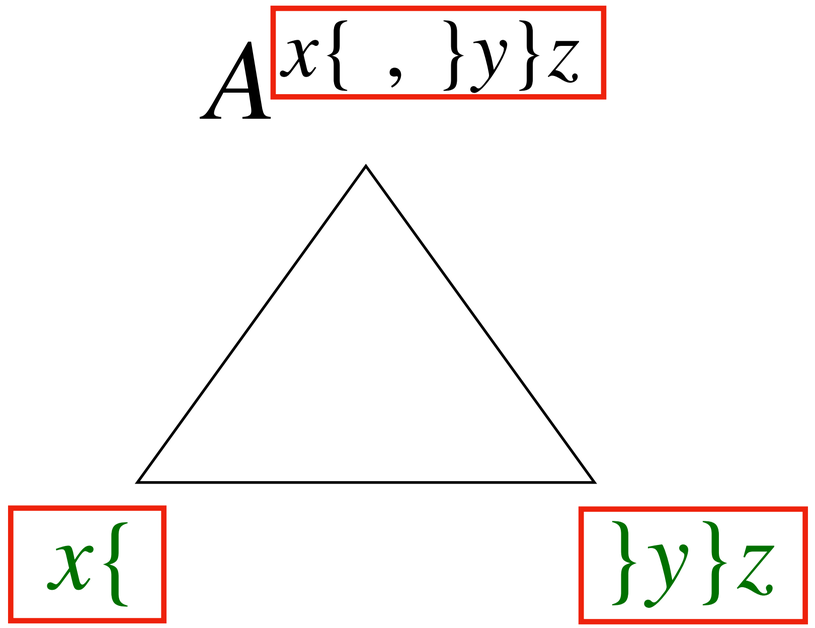
add **pair of indices** that specify substrings of the **input document**



Enumeration Algorithm - Preprocessing 2

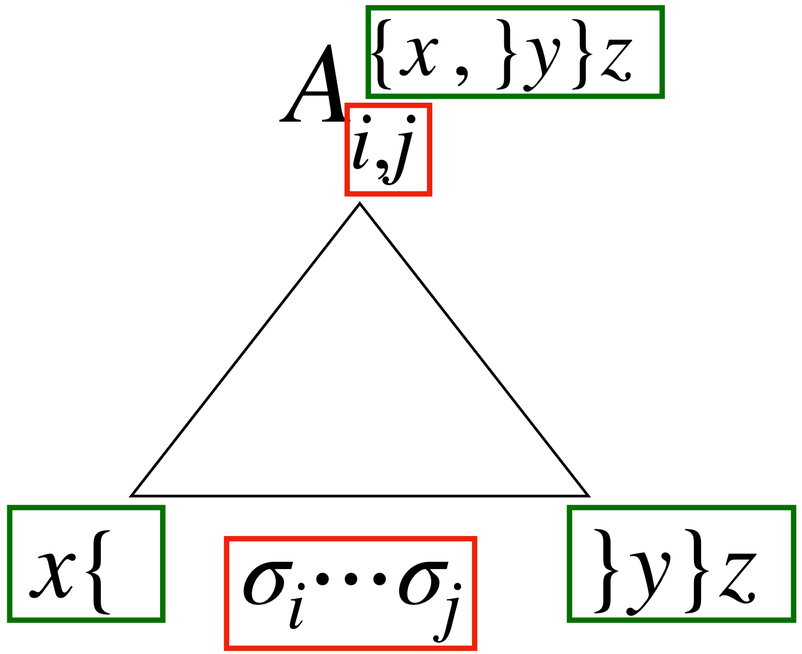
tracking variable operations

add **superscripts** that specify **variable operations**



Enumeration Algorithm - Preprocessing Recap

new enriched non-terminals



Lemma

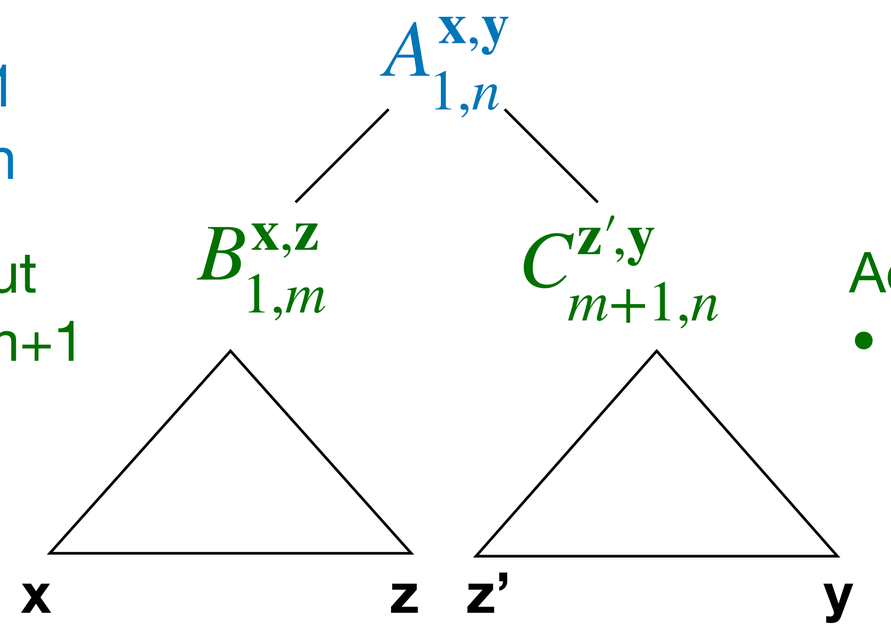
The cost is d^3

Enumeration Algorithm - Actual Enumeration

Recursively builds the output

Add to output
- **x** in position 1
- **y** in position n

Add to the output
- **z** in position m+1



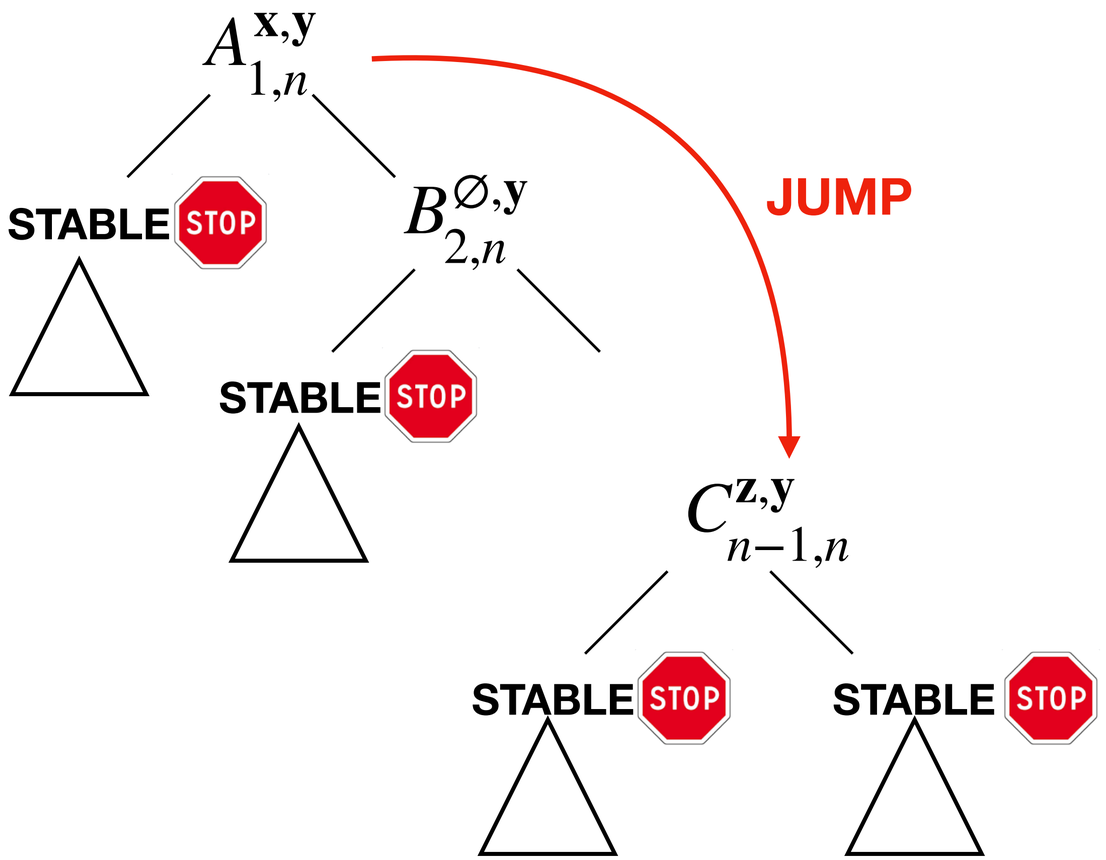
Add to the output
- **z'** in position m+1

Continue recursively with subtrees

Stable non-terminals
no variable operations as descendants

Enumeration Algorithm - Improving the Delay

Depth of stable non-terminals is linear in the document



Lemma

The cost is d^5

Jumping ensures constant delay!

Generalizations

Discuss 3 Generalizations

x	y	z
[1,14⟩		[1,36⟩
[42,47⟩	[52,65⟩	[42,65⟩
[102,110⟩	[115,125⟩	

1. Incomplete spanners

[Maturana, Riveros, Vrgoc 2018]

x	y	z
[1,14⟩	EDBT	
[42,47⟩	Summer	
[102,110⟩	School	

2. Relational spanners

[P, ten Cate, Fagin, Kimelfeld 2019]

x	y	z	weight
[1,14⟩	[30,36⟩	[1,36⟩	0.5
[42,47⟩	[52,65⟩	[42,65⟩	0.9
[102,110⟩	[115,125⟩	[102,125⟩	02

3. Annotating spanners

[Doleschal, Kimelfeld, Martens, P 2019]

Extraction Confidence

- How do we incorporate **confidence** in the extracted tuples?
- We consider an extension of the model, where tuples are annotated with values, e.g.,
  - Real numbers in $[0,1]$
  - Categorical values in $\{\text{low}, \text{medium}, \text{high}\}$
  - Natural numbers
  - …
- More generally, semiring annotations

Commutative Semirings

- $(\mathbf{K}, \mathbb{0}, \mathbb{1}, \oplus, \otimes)$
  - $\oplus$ is associative & commutative with identity $\mathbb{0}$
  - $\otimes$ is associative & commutative with identity $\mathbb{1}$
  - $\otimes$ distributes over $\oplus$
  - $\mathbb{0}$ is absorbing for $\otimes$: $\mathbb{0} \otimes a = \mathbb{0}$
- Examples:

Counting semiring: $(\mathbb{N}, 0, 1, +, \times)$

Probability semiring: $(\mathbb{R}_{\geq 0}, 0, 1, +, \times)$

Boolean semiring: $(\{\mathbf{T}, \mathbf{F}\}, \mathbf{T}, \mathbf{F}, \vee, \wedge)$

Tropical semiring: $(\mathbb{N} \cup \{\infty\}, \infty, 0, \min, +)$

Annotated Relations

- An **annotated relation** is a relation where *each tuple is assigned a provenance annotation* from a commutative semiring $(\mathbf{K}, \mathbb{0}, \mathbb{1}, \oplus, \otimes)$
- Positive RA incorporates the annotation:

$$R_1 \bowtie R_2 : \{\ (t_1 \bowtie t_2,\ a_1 \otimes a_2)\ \mid (t_1, a_1) \in R_1\ ,\ (t_2, a_2) \in R_2\ \}$$

$$R_1 \cup R_2 : \{\ (t,\ a_1 \oplus a_2)\ \mid\ (t, a_1) \in R_1\ ,\ (t, a_2) \in R_2\ \}$$

$$\pi_A R : \{\ (t,\ \oplus\{a \mid (s, a) \in R\ ,\ \pi_A(s) = t\}\ \}$$

Identify $t \notin R$ with $(t, \mathbb{0}) \in R$

[Green, Karvounarakis, Tannen: Provenance semirings. 2007]

Annotating Spanners

An annotating spanner maps every document into an annotated relation over the document's spans

- We now assume a fixed **finite alphabet** and a fixed **commutative semiring**

Kaspersky Lab CEO Eugene Kaspersky said Intel CEO Paul Otellini and the Intel board had no idea what they were in for when the company announced it was acquiring McAfee on August 19, 2010.

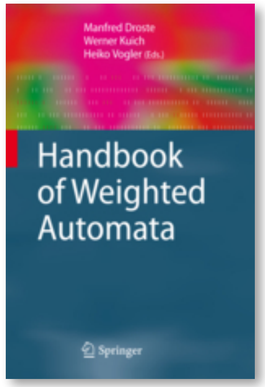
Document d

x	y	z	prov
[1,14⟩	[30,36⟩	[1,36⟩	k_1
[42,47⟩	[52,65⟩	[42,65⟩	k_2
[102,110⟩	[115,125⟩	[102,125⟩	k_3

Annotated relation over the spans of d

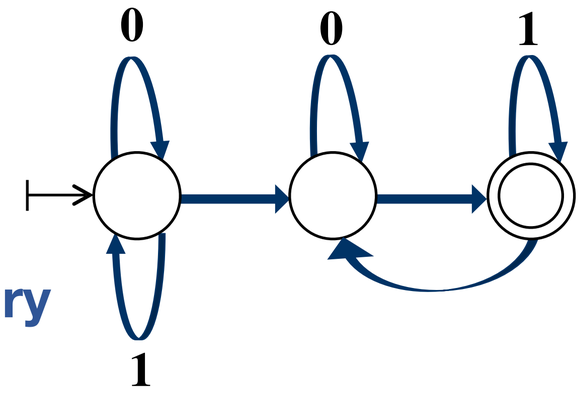
Representation via Weighted Automata

- As an extension of an existing representation, we can use spanner automata weighted by semirings
- In fact, the more appropriate formalism is that of a **weighted finite-state transducer**
  - [Droste, M., Kuich, W. and Vogler, H. eds., 2009. *Handbook of weighted automata*. Springer Science & Business Media.]

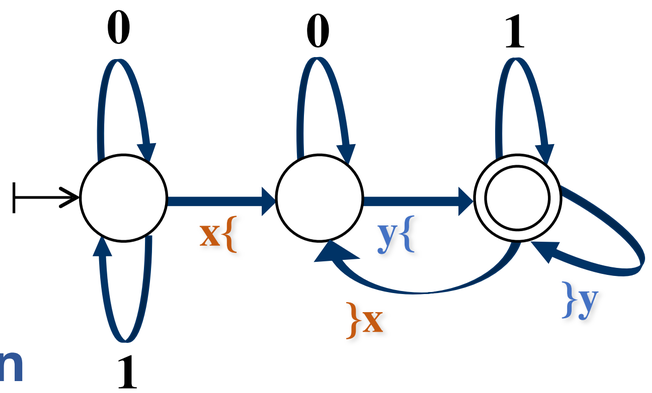


Annotating Spanners as Weighted Automata

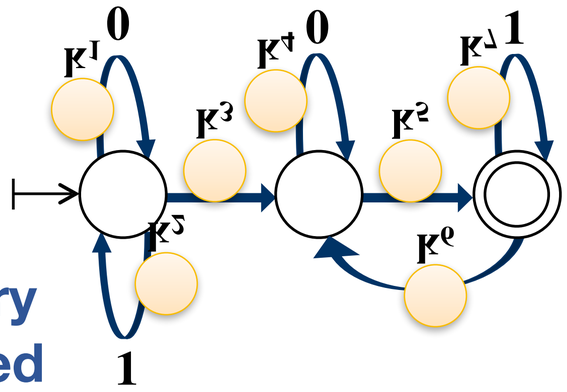
Ordinary NFA



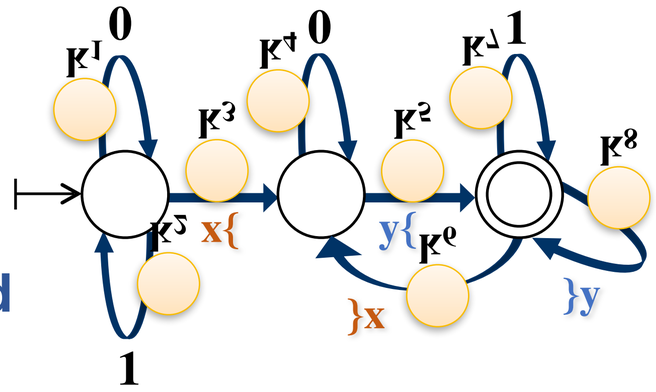
Spanner Automaton



Ordinary weighted NFA



Weighted Spanner Automaton



$$w(d) = \bigoplus \bigotimes w(e)$$

accept $e \in \varrho$

runs ϱ

$$w(d,t) = \bigoplus \bigotimes w(e)$$

t-generating $e \in \varrho$

runs ϱ

Some Results

THM. On every commutative semiring, the class of annotating spanners is closed under union, projection, and natural join.

For the probability/counting semiring
(assuming no epsilon cycles)

THM. [comb./ext. complexity] The weight of a tuple can be computed in polynomial time.

THM. [comb./ext. complexity] It is NP-hard to find a *maximum-weight tuple*, or any sub-exponential approximation thereof.

[Doleschal, K, Martens, Peterfreund 2019]