LFDS - 17/11/2015
UCL, London

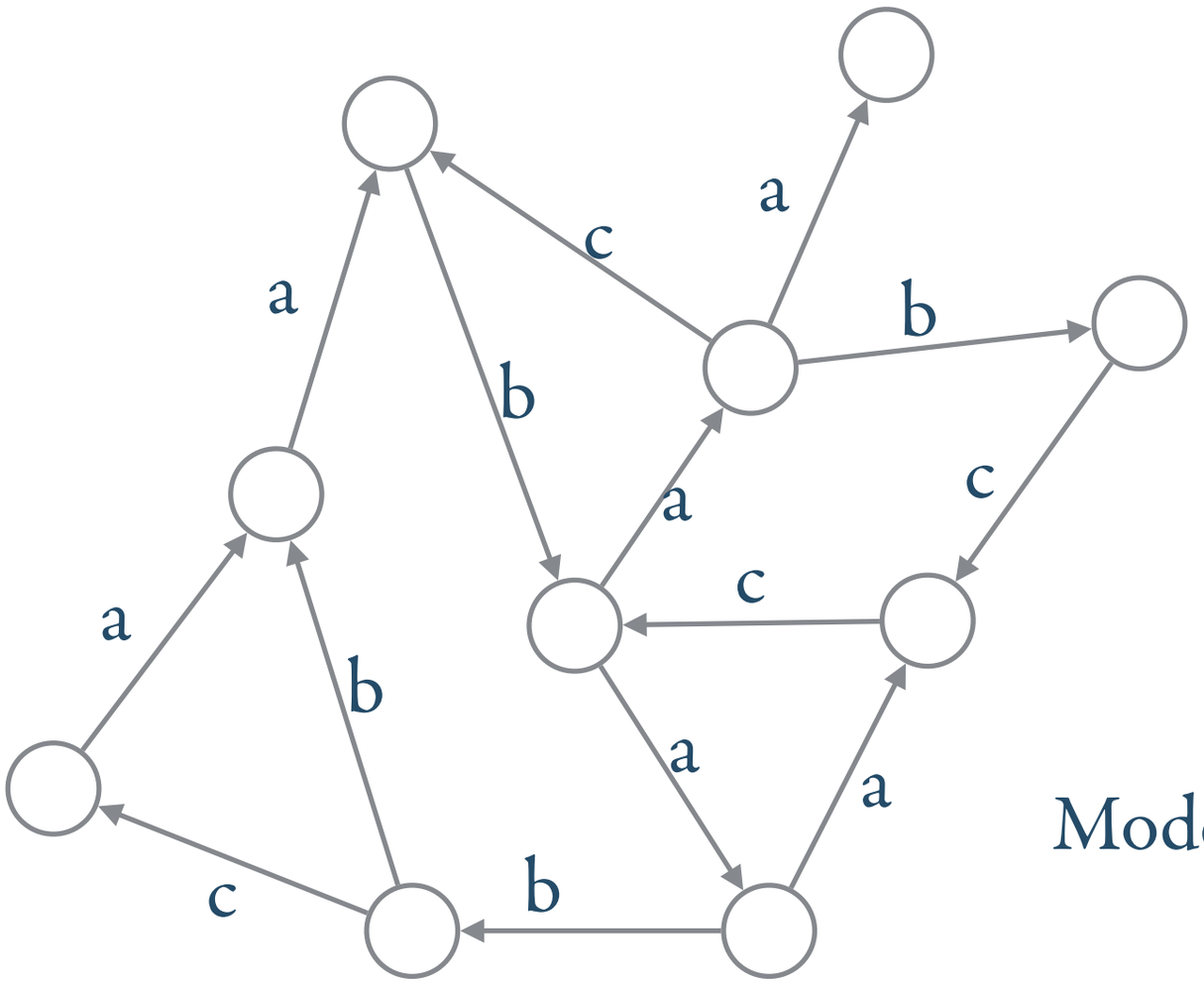
Path Logics for Querying Graphs

combining expressiveness and efficiency

Diego Figueira
CNRS, LaBRI
France

Graph*databases*

Semantic web / RDF / social networks / . . .

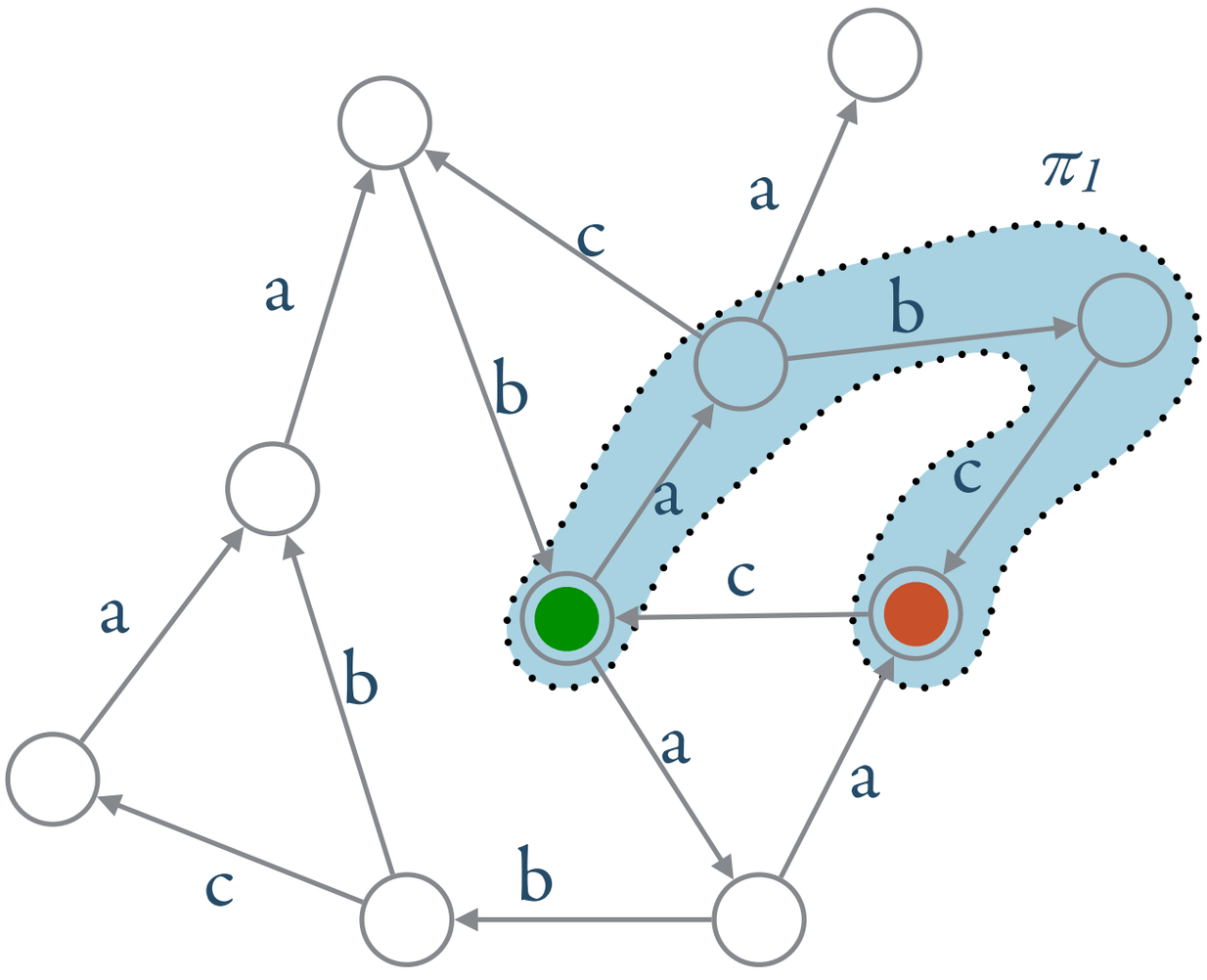


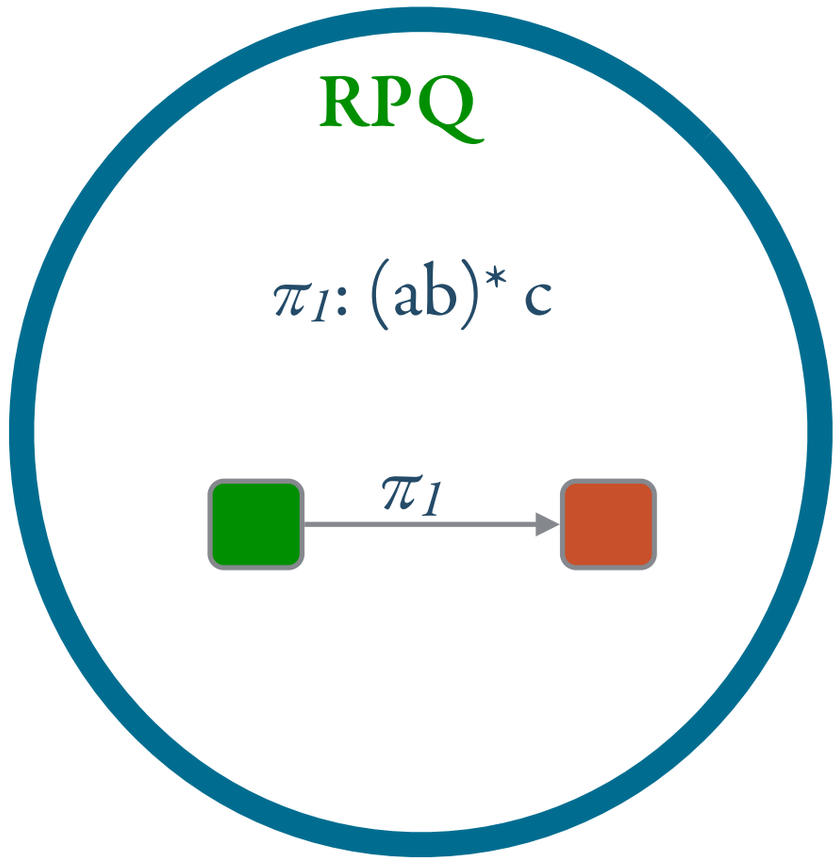
"Entities + Relations"

Modelled as: edge-labelled directed graphs

Notion of ***path*** of central importance

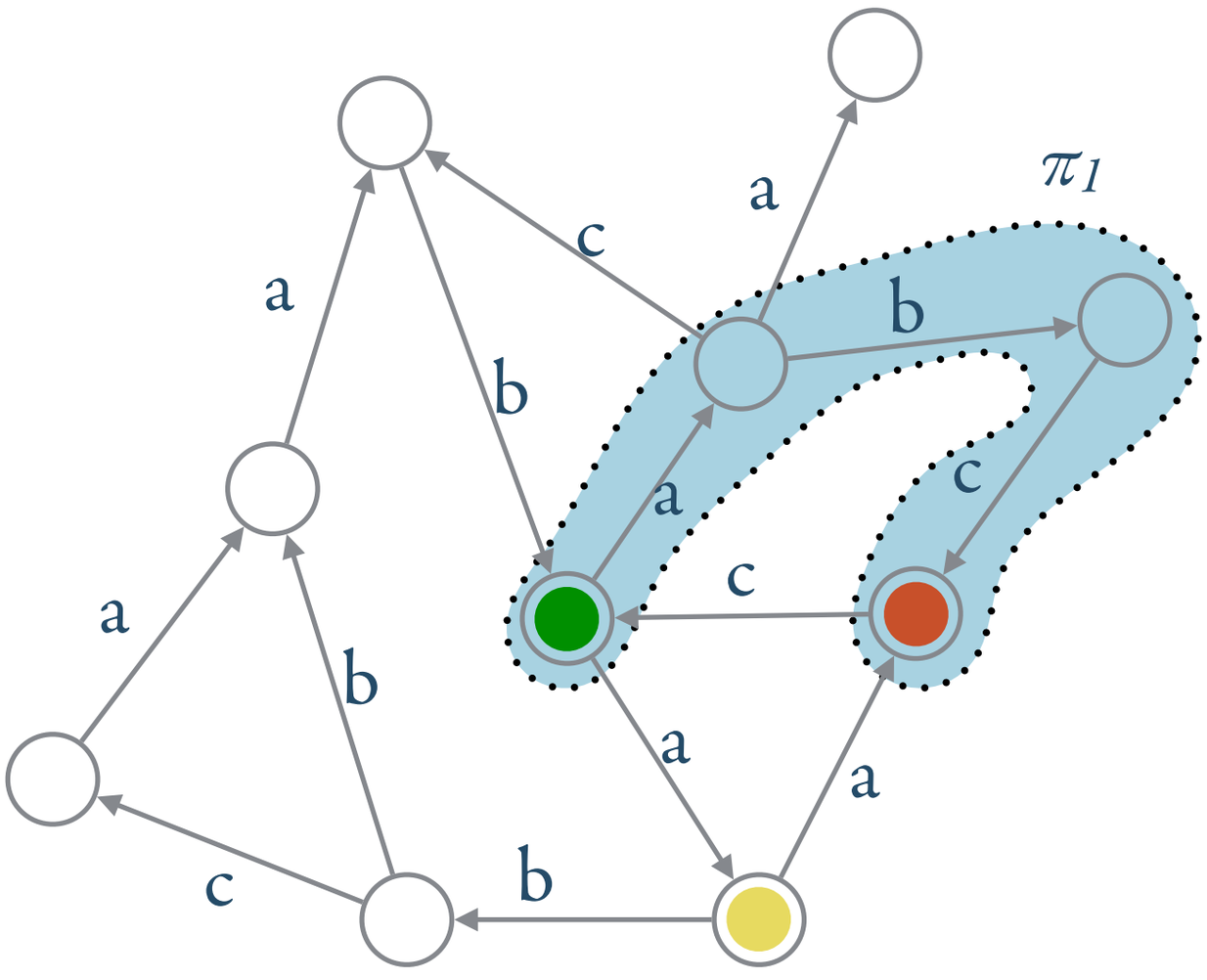
Graph*databases*

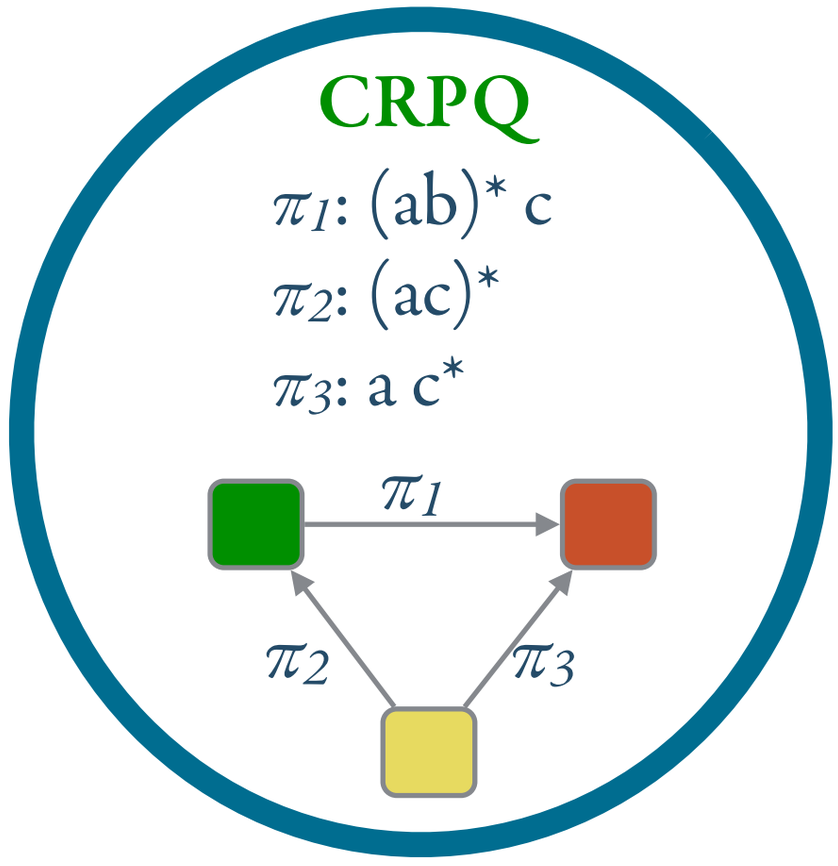




Evaluation: P (combined)
NL (data)

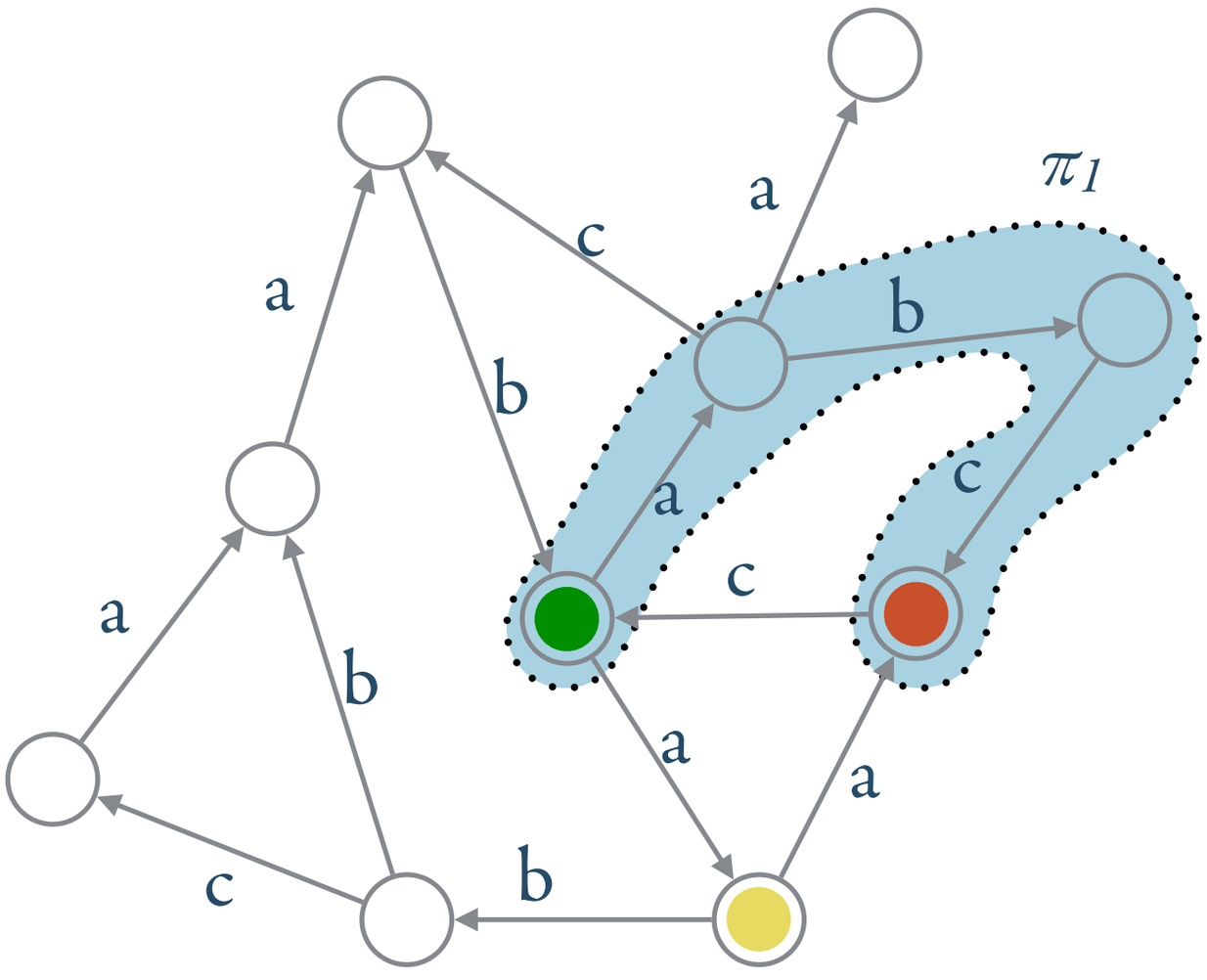
Graph*databases*

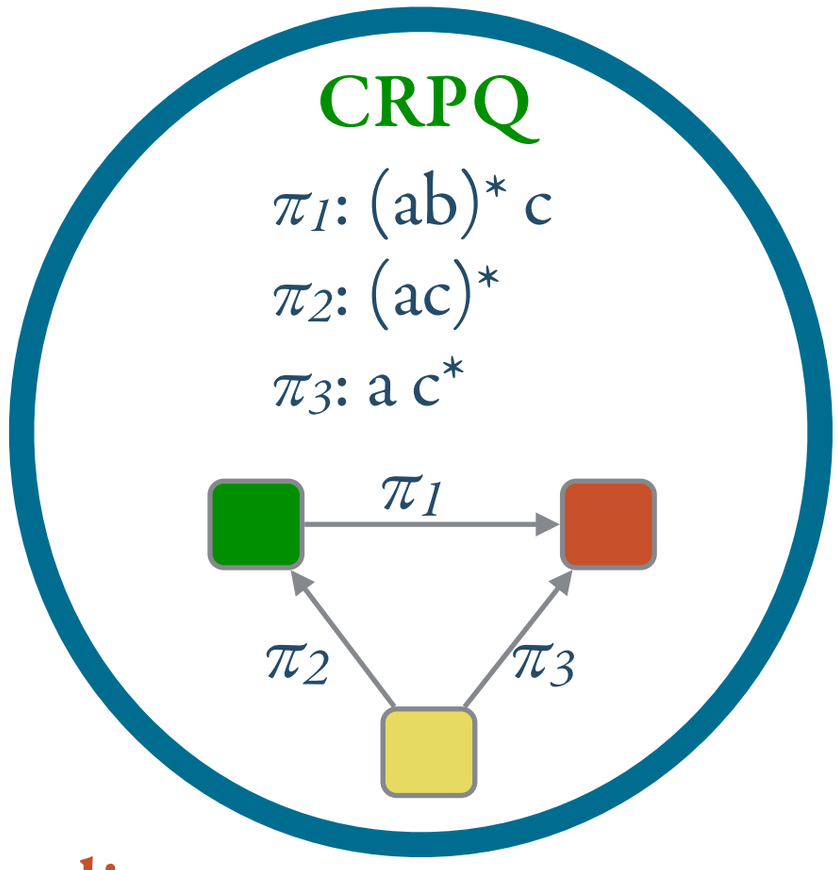




Evaluation: NP (combined)
NL (data)

Graph*databases*





Acyclic

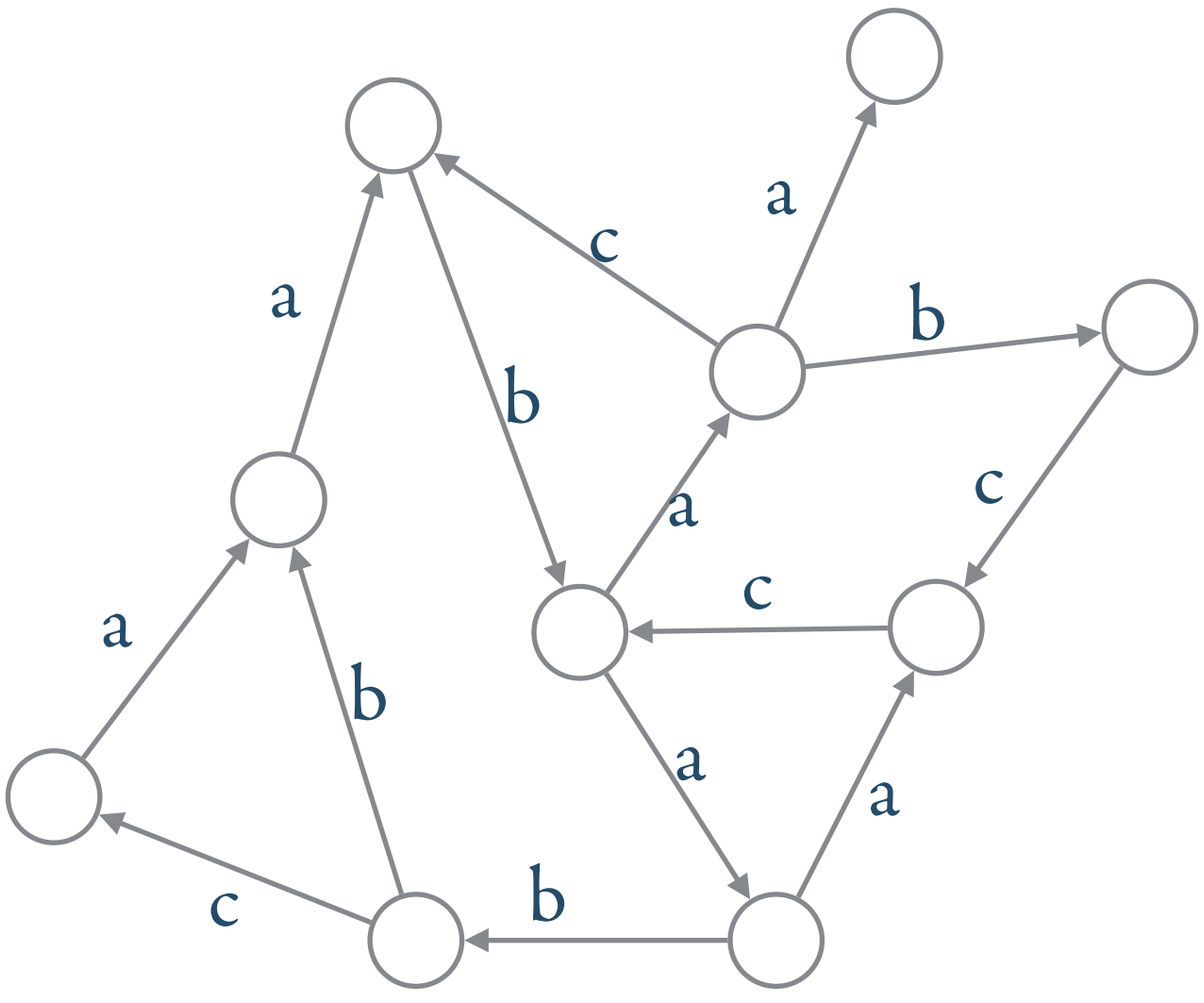
P

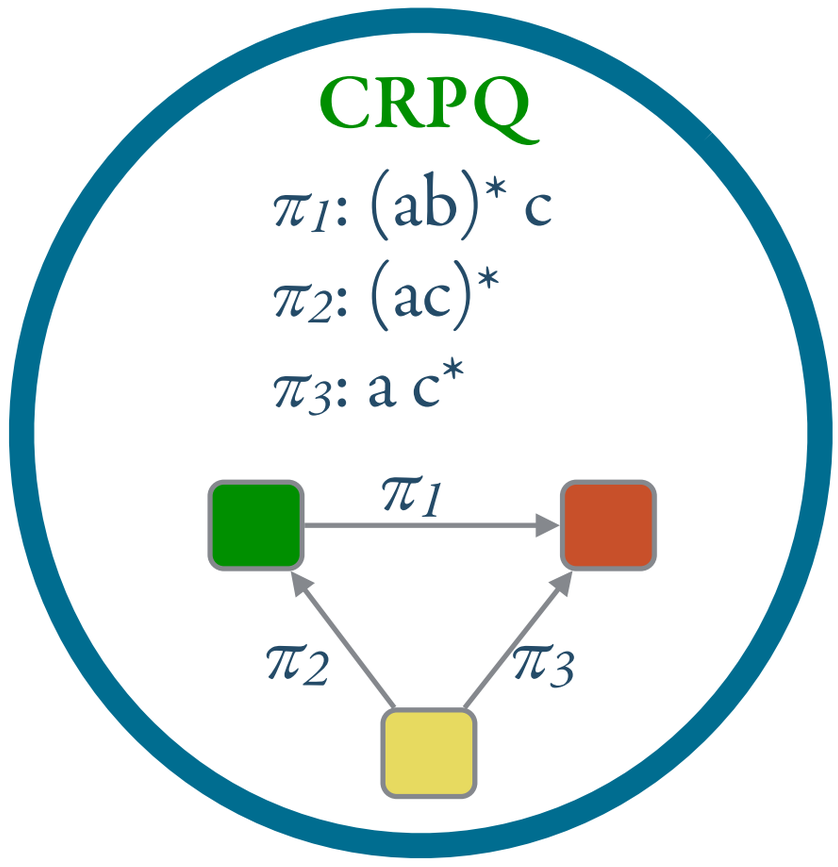
Evaluation: ~~NP~~ (combined)

NL (data)

Unions, inverse

Graph*databases*





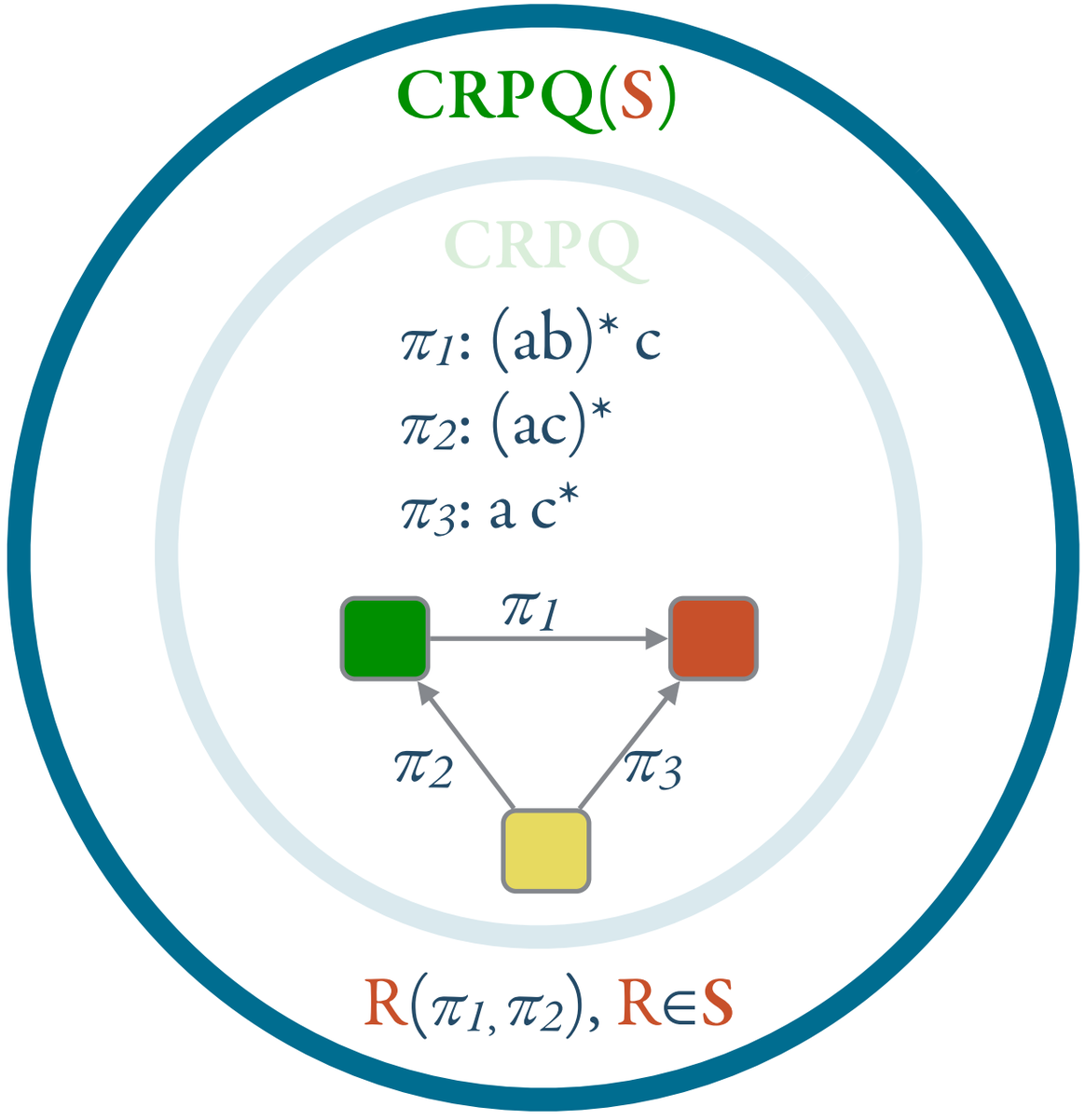
What about...

*"All the pairs (**u**,**v**) that can reach some node **z** in the same number of steps"*

Graph*databases*

What about testing for **relations** on the paths?

- $|\pi_i| = |\pi_j|$
- π_i is a **prefix** of π_j
- π_i is a **subsequence** of π_j
- π_i is a **factor** of π_j
- $\pi_i = \pi_j$ projected onto A



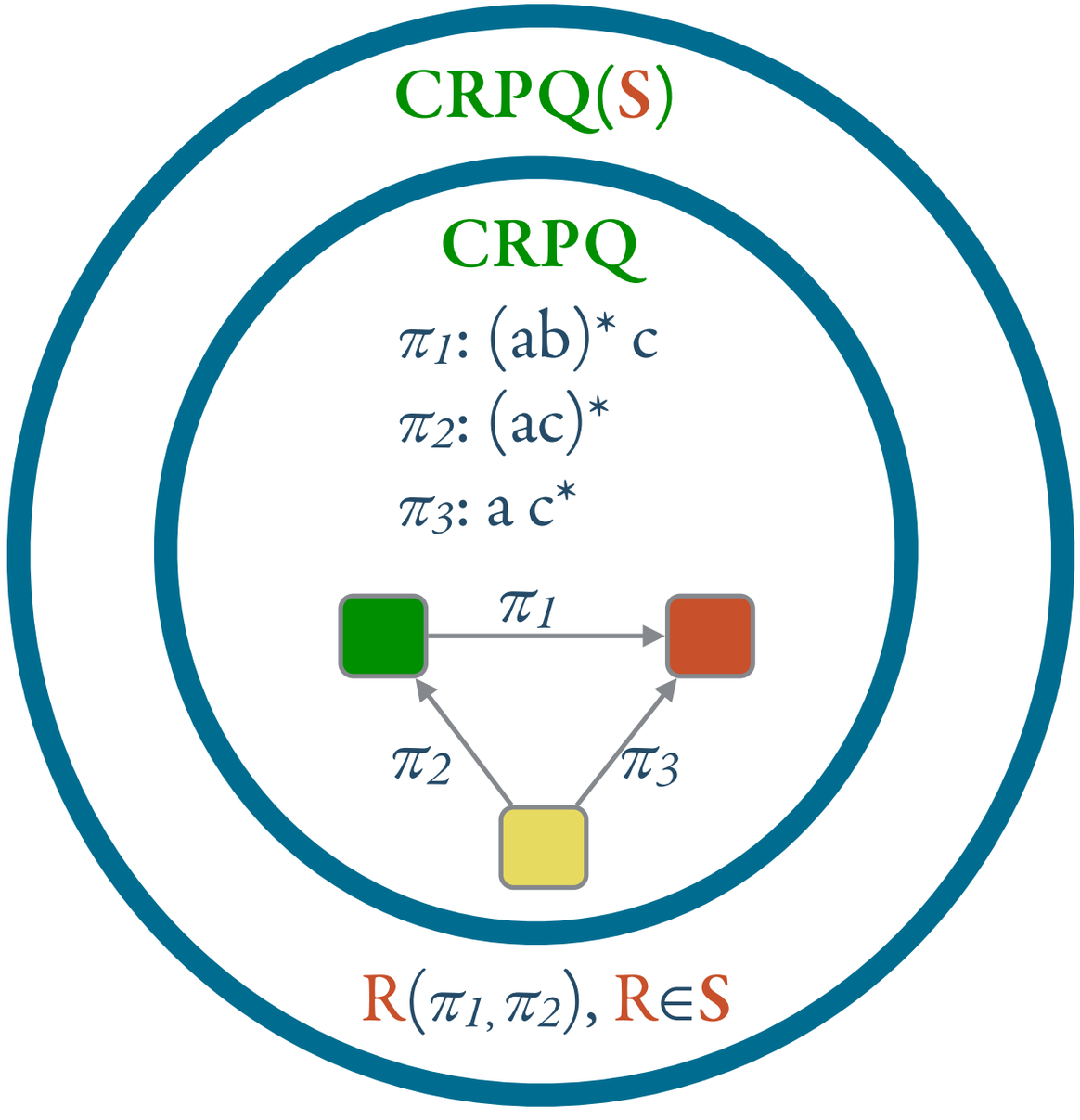
Motivations from: entity resolution, semantic associations, crime detection,...

Graph*databases*

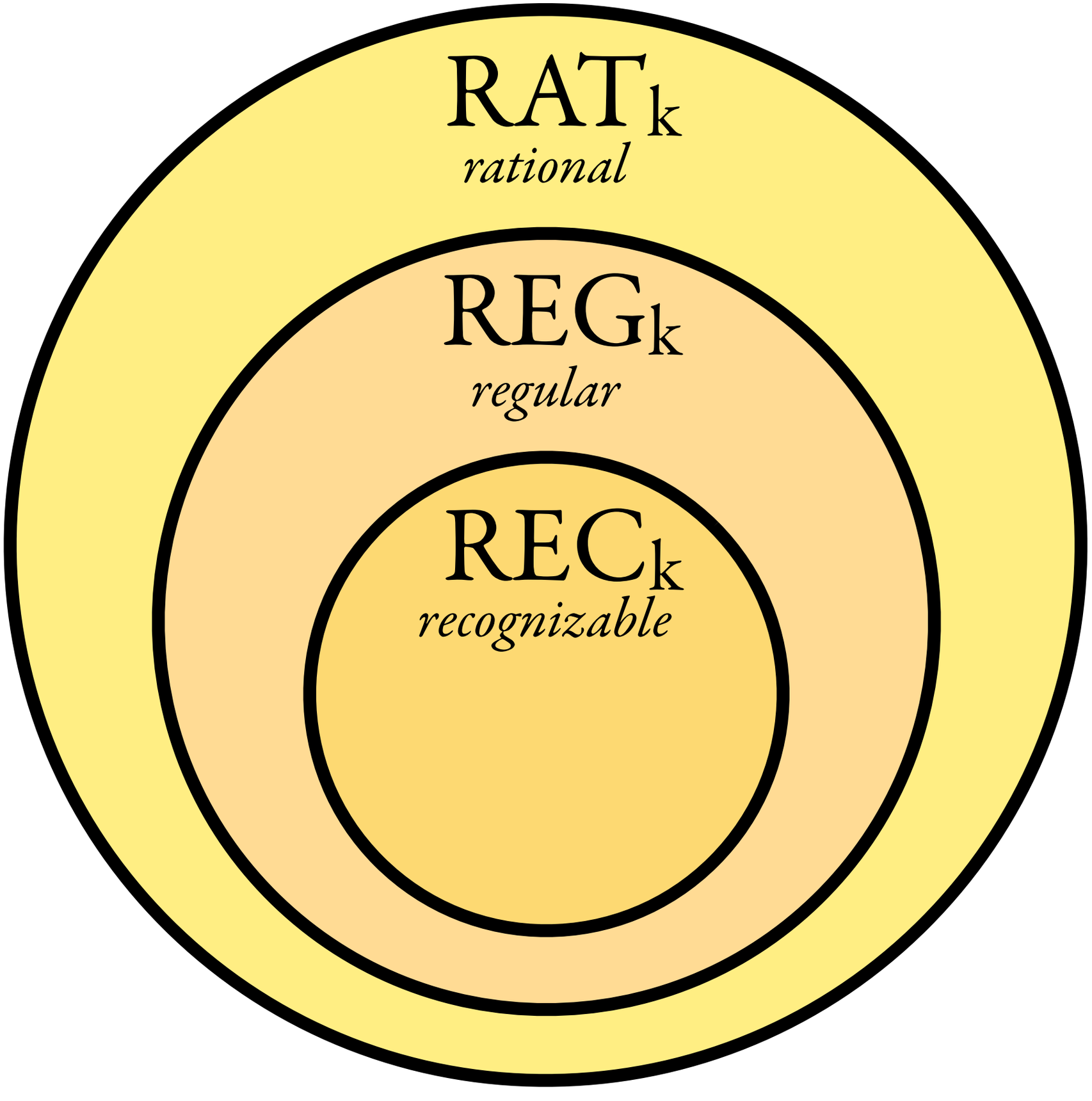
What about testing for **relations** on the paths?

CRPQ(S) = **CRPQ** +
tests $R(\pi_{i_1},\ldots,\pi_{i_n})$, $R \in$ **S**

S: Class of well-behaved word relations...



Word relations

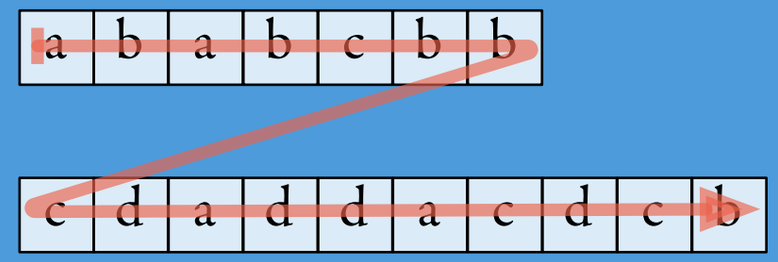


binary relations

$R \subseteq \mathbb{A}^* \times \mathbb{A}^*$

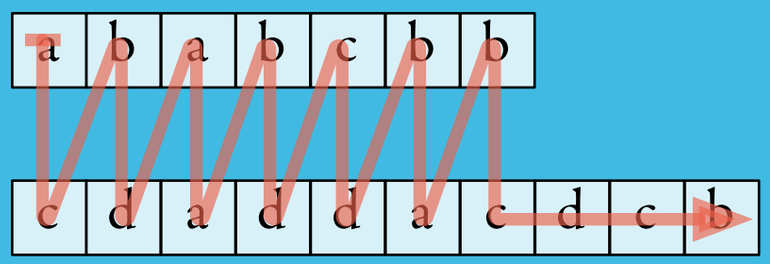
REC_2

recognizable



REG_2

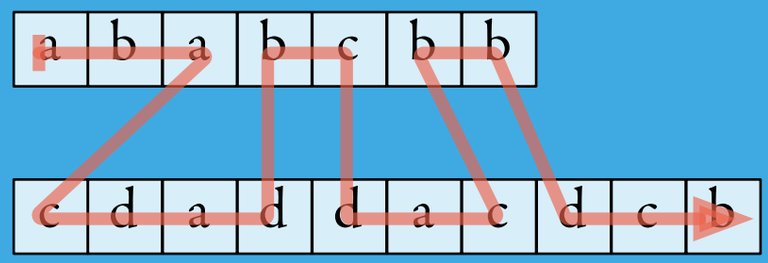
regular



prefix, equal, equal length, ...

RAT_2

rational



suffix, infix, projection, subsequence, ...

Graph *databases*

CRPQ(S) = **CRPQ** +
tests $R(\pi_{i_1},\ldots,\pi_{i_n})$, $R \in$ **S**

CRPQ(REC) NP/NL complexity

Can this be extended?

CRPQ(REG) PSPACE/NL complexity

CRPQ(RAT) *undecidable*

Related to the Intersection Problem:
Given relations $R_1,\ldots,R_n$, whether $R_1 \cap \cdots \cap R_n \neq \emptyset$

intersection *problem*

$R \cap S = \varnothing$ *?*

R, S : classes of binary relations

input: $\mathrm{R} \in \boldsymbol{R}$, $\mathrm{S} \in \boldsymbol{S}$
output: $\mathrm{R} \cap \mathrm{S} = \varnothing$ *?*

intersection *problem*

$R \cap S = \varnothing$ *?*

R, S : classes of binary relations

it has been studied...

$(u_{i_1} \ldots u_{i_n}, v_{i_1} \ldots v_{i_n})$

PCP

REG ∩ RAT = ∅ *?*
already undecidable

input: $R \in R, S \in S$
output: $R \cap S = \varnothing$ *?*

...but what about
real world relations?

intersection *problem*

$R \cap S = \varnothing$ *?*

R, S : classes of binary relations

it has been studied...

REG ∩ RAT = ∅ *?*
already undecidable

input: R ∈ R, S ∈ S
output: R ∩ S = ∅ *?*

...but what about
real world relations?

like

subsequence...?

subword...?

suffix...?

intersection *problem*

$R \cap S = \varnothing$ *?*

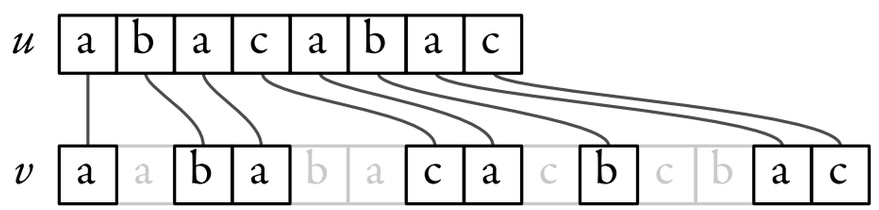
R, S : classes of binary relations

it has been studied...

REG ∩ RAT = ∅ *?*
already undecidable

input: $R \in R, S \in S$
output: $R \cap S = \varnothing$ *?*

$u \sqsubseteq v$



subsequence

subsequence...?

suffix...?

subword...?

Can we extend CRPQ beyond REG relations?

Language	Data complexity	Combined complexity
CRPQ(REG_k)	NL	PSPACE
CRPQ(RAT_k)	Undecidable	Undecidable
CRPQ(REG_k + suffix)	Undecidable	Undecidable
CRPQ(REG_k + factor)	Undecidable	Undecidable
CRPQ(REG_k + subsequence)	non-elementary	non-PR
CRPQ(suffix)	NL	PSPACE
CRPQ(factor)	PSPACE	PSPACE
CRPQ(subsequene)	PSPACE	NEXPTIME

$\forall\ k>1$

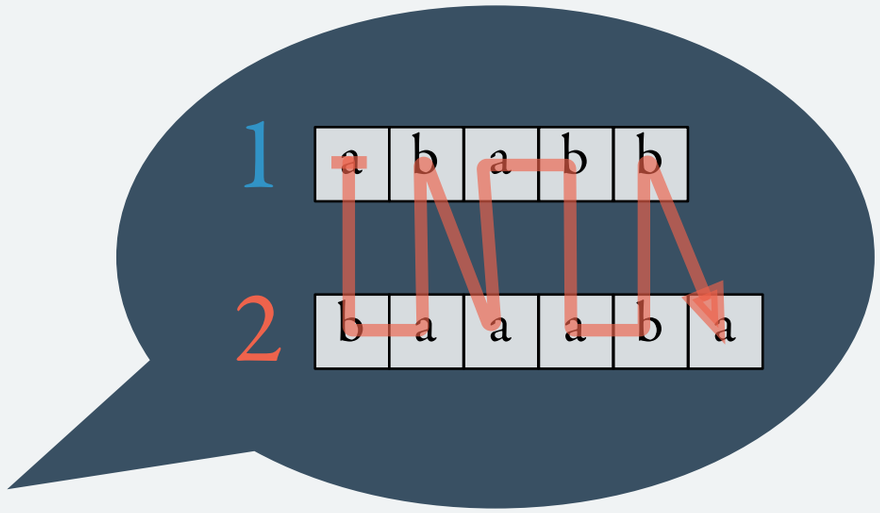
Can we extend CRPQ beyond REG relations?

Proposed alternative: **approximate** RAT through REG + counters

How?

1) take a an NFA
2) add counters
3) use it to read k-tuples of words

2 tapes over $\mathbb{A}$ $\approx$ 1 tape over $\mathbb{A}\times\{1,2\}$



1 2 2 1 2 1 1 2 2 1 2

a b a b a a b a b b a

$\in$

$(\mathbb{A}\times\{1,2\})^*$

$$2 \text{ tapes over } \mathbb{A} \;\approx\; 1 \text{ tape over } \mathbb{A}\times\{1,2\}$$

$$\left[\!\!\left[\begin{array}{cccccccccccc} 1 & 2 & 2 & 1 & 2 & 1 & 1 & 2 & 2 & 1 & 2 \\ a & b & a & b & a & a & b & a & b & b & a \end{array} \right]\!\!\right] = (\text{ababb}, \quad)$$

$$\in \qquad\qquad\qquad \in$$

$$(\mathbb{A}\times\{1,2\})^* \qquad\qquad \mathbb{A}^*\times\mathbb{A}^*$$

$$2 \text{ tapes over } \mathbb{A} \;\approx\; 1 \text{ tape over } \mathbb{A}\times\{1,2\}$$

$$\left[\!\!\left[\begin{array}{cccccccccccc} 1 & 2 & 2 & 1 & 2 & 1 & 1 & 2 & 2 & 1 & 2 \\ a & b & a & b & a & a & b & a & b & b & a \end{array} \right]\!\!\right] = \Big(\text{ababb} \,,\, \text{baaaba} \Big)$$

$$\Cap \qquad\qquad\qquad \Cap$$

$$(\mathbb{A}\times\{1,2\})^* \qquad\qquad \mathbb{A}^*\times\mathbb{A}^*$$

$$2 \text{ tapes over } \mathbb{A} \;\approx\; 1 \text{ tape over } \mathbb{A}\times\{1,2\}$$

control word

$$\left[\!\!\left[\begin{array}{cccccccccccc} 1 & 2 & 2 & 1 & 2 & 1 & 1 & 2 & 2 & 1 & 2 \\ a & b & a & b & a & a & b & a & b & b & a \end{array}\right]\!\!\right] = (\text{ababb}, \text{baaaba})$$

$(1|2)^*$-controlled

$$[\![(\mathbb{A}\times\{1,2\})^*]\!] = \mathbb{A}^*\times\mathbb{A}^*$$

$$[\![((a,1)(a,2)|(b,1)(b,2))^*]\!] = \text{equality}$$

$(12)^*$-controlled

$$2 \text{ tapes over } \mathbb{A} \;\approx\; 1 \text{ tape over } \mathbb{A}\times\{1,2\}$$

control word

$$\left[\!\!\left[\begin{array}{cccccccccccc} 1 & 2 & 2 & 1 & 2 & 1 & 1 & 2 & 2 & 1 & 2 \\ a & b & a & b & a & a & b & a & b & b & a \end{array} \right]\!\!\right] = (\text{ababb}, \text{baaaba})$$

$(1|2)^*$-controlled

$$[\![(\mathbb{A}\times\{1,2\})^*]\!] = \mathbb{A}^*\times\mathbb{A}^*$$

$$[\![((a,1)(a,2)|(b,1)(b,2))^*]\!] = \text{equality}$$

$(12)^*$-controlled

$L\subseteq\{1,2\}^*$

$$\mathbf{Rel}(L) = \{[\![S]\!] \mid S\in \mathrm{REG}(\mathbb{A}\times\{1,2\}) \text{ is } L\text{-controlled}\}$$

Eg:

$$\mathbf{Rel}((1|2)^*) = \mathrm{RAT}_2$$

$$\mathbf{Rel}((12)^*(1^*|2^*)) = \mathrm{REG}_2$$

$$\mathbf{Rel}(1^*2^*) = \mathrm{REC}_2$$

$$\mathbf{Rel}((12)^*) = \text{length-preserving } \mathrm{REG}_2$$

$$\mathbf{Rel}((1^*|2^*)(12)^*) = \mathrm{REG}_2^{rev}$$

Approximate with regular relations that can **count patterns**

$$R = \{ (u, v) \mid \begin{array}{c} \#\text{ of times } (ab)^*c \text{ appears in } u \\ = \\ 2 \cdot \#\text{ of times } c^*b \text{ appears in } v \end{array} \}$$

More than just counting letters

Idea

Instead of **regular languages**...

$$\mathbf{Rel}(L) = \{ [\![S]\!] \mid S \in \mathbf{REG}(\mathbb{A} \times \{1,2\}) \text{ is } L\text{-controlled} \}$$

...use **automata with counting**

Idea

Evaluation of CRPQ with counting is feasible

PSPACE in combined complexity

NL in data complexity

Parikh Automata*

[Klaedtke & Rueß]

NFA with **n** counters $c_1,\dots,c_n$ and a semilinear set $S \subseteq \mathbb{N}^{\mathbf{n}}$

$$(\mathbb{A},Q,q_0,\delta,F,\mathbf{n},\mathbf{S})$$

Transitions of δ: $(q,a,(x_1,\dots,x_n),q') \in Q \times \mathbb{A} \times \mathbb{N}^{\mathbf{n}} \times Q$

Run:

- Initial configuration: $(q_0,(0,\dots,0)) \in Q \times \mathbb{N}^{\mathbf{n}}$
- $(q,\mathbf{x}) \xrightarrow{(q,a,y,p)\ \in\ \delta} (p,(\mathbf{x}+\mathbf{y}))$
- Acceptance: last configuration in $F \times \mathbf{S}$

* Many equivalent definitions (eg. reversal-bounded counter systems)

Parikh Automata*

[Klaedtke & Rueß]

dimension

NFA with **n** counters $c_1,\dots,c_n$ and a semilinear set $S \subseteq \mathbb{N}^{\mathbf{n}}$

$(\mathbb{A},Q,q_0,\delta,F,\mathbf{n},\mathbf{S})$

Transitions of δ: $(q,a,(x_1,\dots,x_n),q') \in Q \times \mathbb{A} \times \mathbb{N}^{\mathbf{n}} \times Q$

Run:

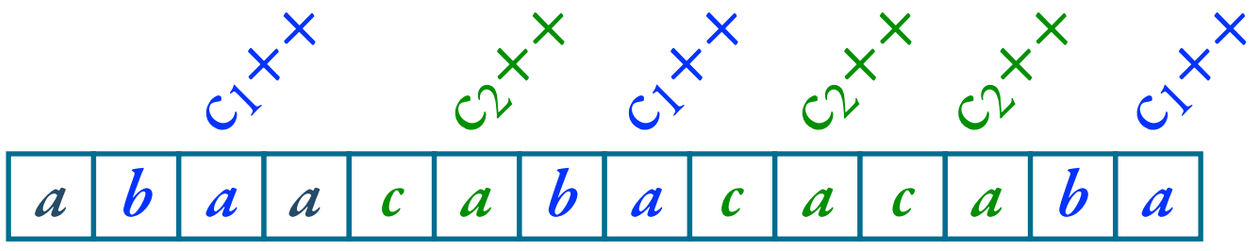
- Initial configuration: $(q_0,(0,\dots,0)) \in Q \times \mathbb{N}^{\mathbf{n}}$
- $(q,\mathbf{x}) \xrightarrow{(q,a,y,p) \in \delta} (p,(\mathbf{x}+\mathbf{y}))$
- Acceptance: last configuration in $F \times \mathbf{S}$

counters can only be **incremented**

* Many equivalent definitions (eg. reversal-bounded counter systems)

Parikh Automata

Eg: $L_{ba=ca} = \{ w \mid \text{number of } a\text{'s after a } b = \text{number of } a\text{'s after a } c \}$



Parikh Automaton $A = (\mathbb{A}, Q, q_0, \delta, F, 2, \{(k,k) \mid k \in \mathbb{N}\})$

- dimension 2 (2 counters)
- increment $\mathbf{c_1}$ whenever we see "**ba**"
- increment $\mathbf{c_2}$ whenever we see "**ca**"
- F=Q
- Semilinear set assures that counters must be equal to accept a word

Parikh Automata

Decidable

non-emptiness,

membership

Closed under

intersection,

union,

(inverse) homomorphisms,

concatenation

(not complementation/iteration)

$$\mathbf{Rel}^{\mathrm{PA}}(L) = \{\, [\![S]\!] \mid S \in \mathbf{PA}(\mathbb{A} \times \{1,2\}) \text{ is } L\text{-controlled} \,\}$$

PA relations

Eg:

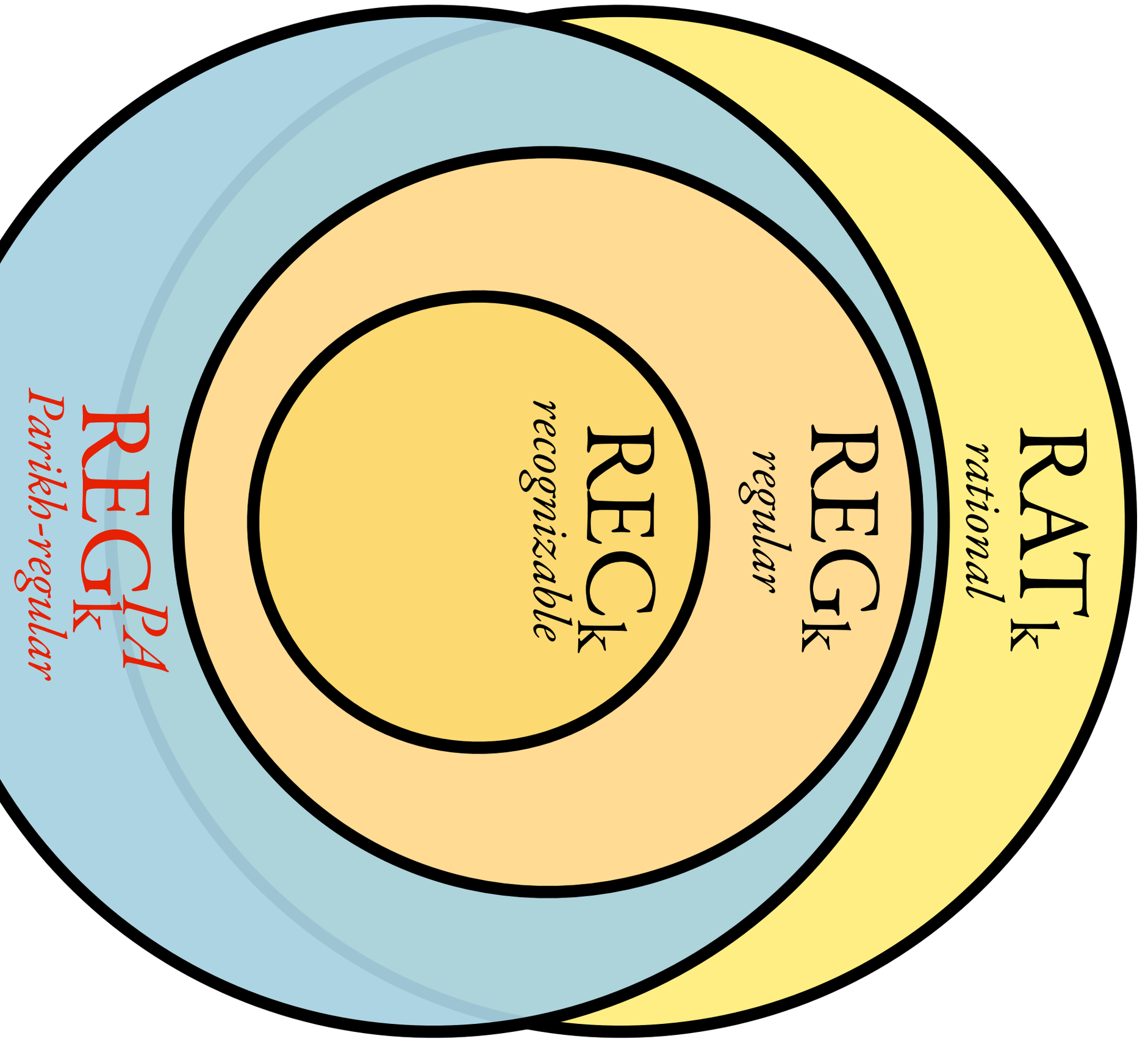
$$\mathrm{REG}_2^{\mathbf{PA}} = \mathbf{Rel}^{\mathbf{PA}}((12)^*(1^*|2^*))$$

$$\mathrm{REG}_{2\,rev}^{\mathbf{PA}} = \mathbf{Rel}^{\mathbf{PA}}((1^*|2^*)(12)^*)$$

$$\mathrm{RAT}_2^{\mathbf{PA}} = \mathbf{Rel}^{\mathbf{PA}}((1|2)^*)$$

$$\vdots$$

Word relations



Theorem: Evaluation of CRPQ(**REG**^{PA}) is
PSPACE in combined complexity
NL in data complexity

Proof ingredients:

- **Intersection problem** for Parikh Automata

  Given PA's $A_1,\ldots,A_n$, is $L(A_1) \cap \cdots \cap L(A_n) \neq \emptyset$?

  is PSPACE-complete

- **Intersection closure** for REG^{PA}

  For all $R,S \in REG^{PA}$, $R \cap S \in REG^{PA}$

  it suffices to intersect the automata representing them

- **Closure under product** of REG^{PA}

Theorem: Evaluation of $CRPQ^{PA}$ (no relations) is
NP in combined complexity
NL in data complexity

Approximating rational relations

$u \sim_k v$ are **k-similar** iff for all w with $|w| \leq k$, they have the same number of appearances of w (as factor)

Given $R \in RAT$,

$$R_k = \{(u,v) \mid u \sim_k u', v \sim_k v', (u', v') \in R\} \in REG^{PA}$$

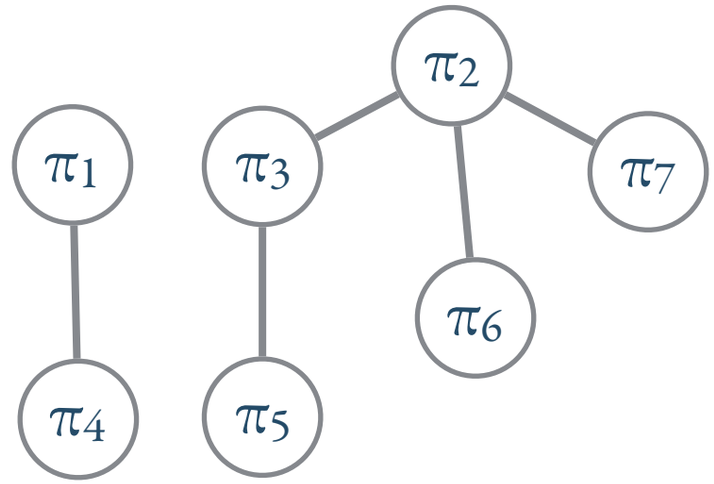
Approximating rational relations

$u \sim_k v$ are **k-similar** iff for all w with $|w| \leq k$, they have the same number of appearances of w ~~(as factor)~~ (as subsequence)

Given $R \in RAT$,

$$R_k = \{(u,v) \mid u \sim_k u', v \sim_k v', (u', v') \in R\} \in REG^{PA}$$

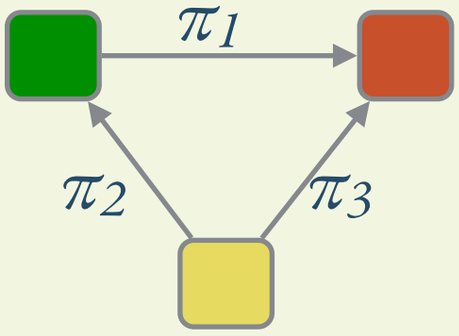
Alternative: Syntactic restrictions



Gaifman multi-graph of path variables

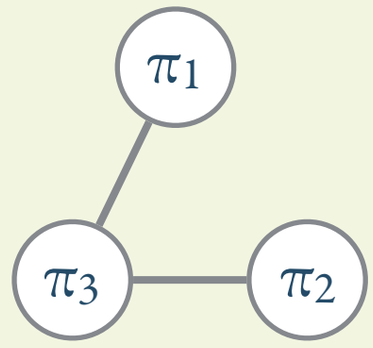
Theorem: Evaluation of **acyclic**-CRPQ(**RAT**$^{\text{PA}}$) is
PSPACE in combined complexity
NL in data complexity

E.g.



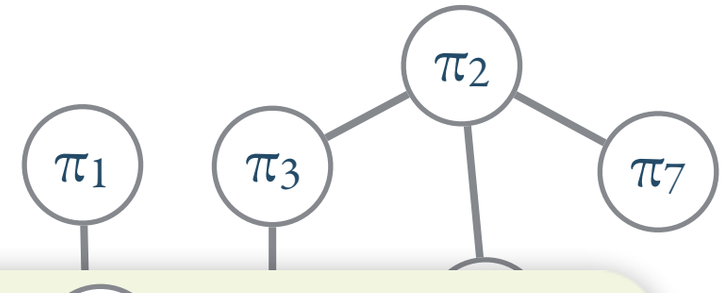
π_1: (ab)* c
π_2: (ac)*
π_3: a c*

R(π_1,π_3)
S(π_3,π_2)

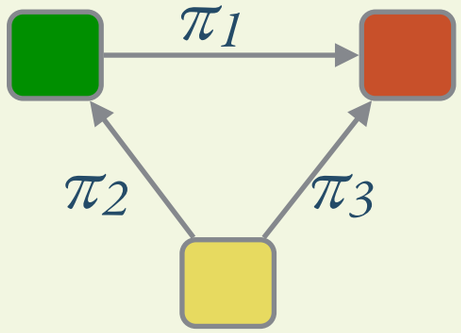


acyclic

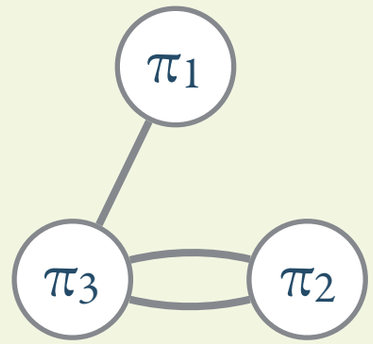
Alternative: Syntactic restrictions



E.g.



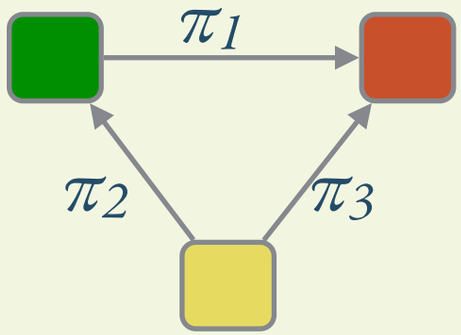
π_1: (ab)* c R(π_1,π_3)
π_2: (ac)* S(π_3,π_2)
π_3: a c* R(π_3,π_2)



cyclic

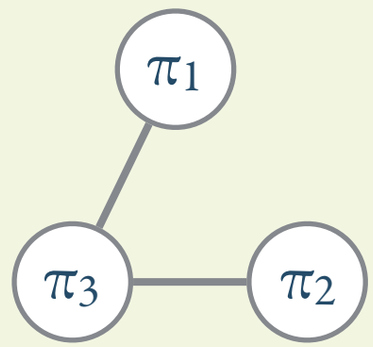
Theorem: Evaluation of **acyclic**-CRPQ(**RAT**^{PA}) is
PSPACE in combined complexity
NL in data complexity

E.g.



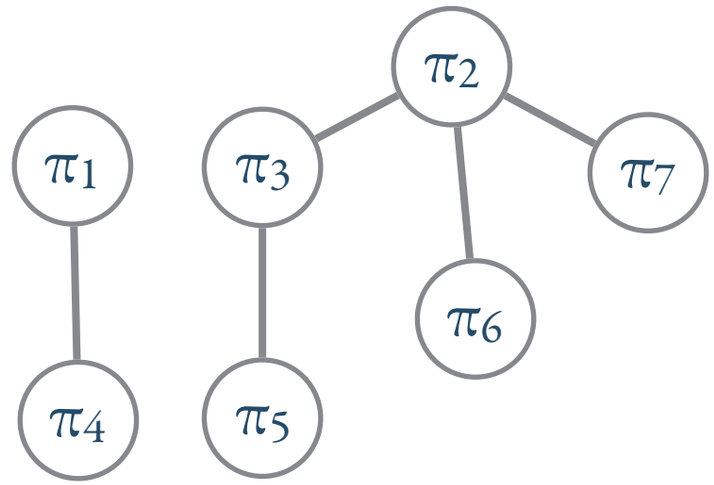
π_1: (ab)* c
π_2: (ac)*
π_3: a c*

R(π_1,π_3)
S(π_3,π_2)



acyclic

Alternative: Syntactic restrictions

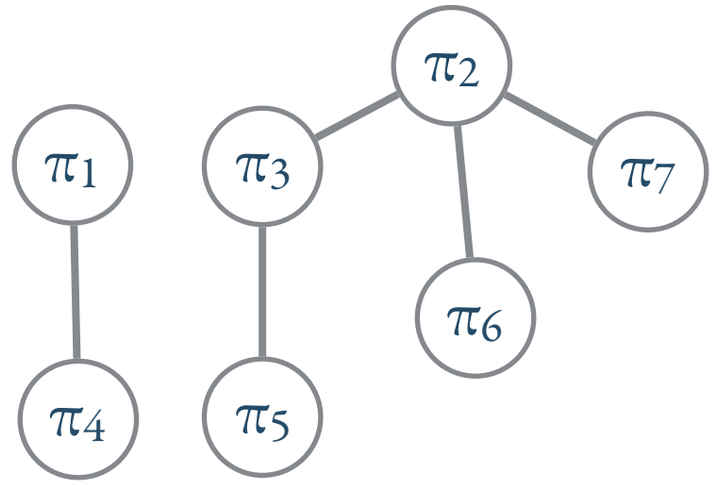


Gaifman multi-graph of path variables

Theorem: Evaluation of **acyclic**-CRPQ(**RAT**$^{\text{PA}}$) is
PSPACE in combined complexity
NL in data complexity

If also fixed **join size**: **NP** combined complexity

Alternative: Syntactic restrictions



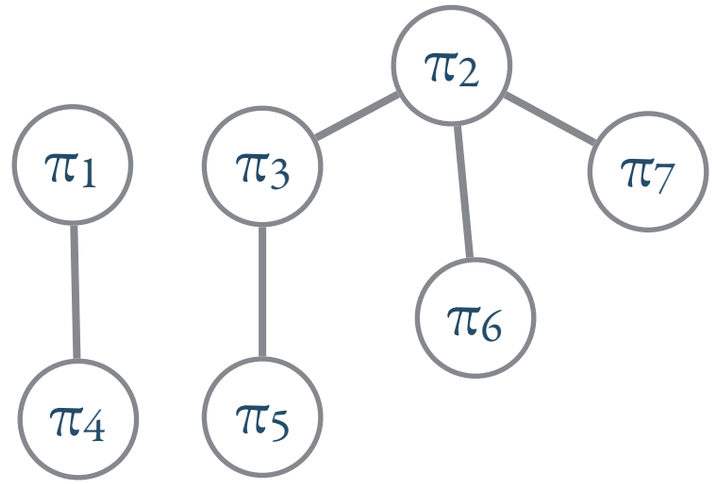
Gaifman multi-graph of path variables

Maximum cardinality of connected component

Theorem: Evaluation of **acyclic**-CRPQ(**RAT**$^{\text{PA}}$) is
PSPACE in combined complexity
NL in data complexity

If also fixed **join size**: **NP** combined complexity

Alternative: Syntactic restrictions



Gaifman multi-graph
of path variables

Theorem: Evaluation of **acyclic**-CRPQ(**RAT**$^{\text{PA}}$) is
PSPACE in combined complexity
NL in data complexity

If also fixed **join size**: **NP** combined complexity

If also fixed **PA dimension** and **unary representation**:
PTIME combined complexity

Conclusion

Counting does not increase complexity

Avoid the curse of of rational relations

Approximating by regular relations with counting

Or staying away from cycles in path relations

Thank you