Graph Queries and Description Logics

From Unrestricted to Finite Entailment

Filip Murlak

PhD Open 2024 @ MIM UW

Graphs

We work with labelled graphs, modelled as relational structures:

- unary predicates = node labels = concept names $A, B, \dots$
- binary predicates = egde labels = role names $r, s, \dots$

That is,

- nodes have multiple labels;
- edges have single labels;
- parallel edges with different labels are allowed;
- in a subgraph, nodes may have fewer labels.

Queries

- Conjunctive queries (CQs), unions of CQs (UCQs)

$$\exists x\,\exists y\; A(x) \wedge r(x,y) \wedge \bar{A}(y) \quad \vee \quad \exists x\,\exists y\,\exists z\; r(x,y) \wedge r(y,z) \wedge r(z,x)$$

  The core of relational query languages, such as SQL.

- Conjunctive regular path queries (CRPQs), unions of CRPQs (UCRPQs)

$$\exists x\; r^{+}(x,x) \quad \vee \quad \exists x\,\exists y\; A(x) \wedge (r^{*} \cup s)(x,y) \wedge (p \cdot B)^{*}(y,x)$$

  Graph query languages have reachability/RPQs at the core (SPARQL, Neo4j's Cypher, SQL/PGQ, GQL ISO Standard under construction).

Description Logics

Basic description logic $\mathcal{ALC}$ has statements of the form

$$C \sqsubseteq D$$

where C, D are *complex concepts* build according to the following grammar by

$$C, D ::= \bot \mid \top \mid A \mid C \sqcup D \mid C \sqcap D \mid \neg C \mid \exists r.C \mid \forall r.C\,.$$

For example, $\mathsf{Person} \sqsubseteq \exists\,\mathsf{childOf}.\,\mathsf{Person}$ and $\mathsf{Male} \sqcap \exists\,\mathsf{childOf}.\,\mathsf{Person} \sqsubseteq \mathsf{Son}$.

$\mathcal{ALC}$ in normal form:

$$\begin{aligned} A_1 \sqcap A_2 \sqcap \cdots \sqcap A_n &\sqsubseteq B_1 \sqcup B_2 \sqcup \cdots \sqcup B_m \\ A &\sqsubseteq \exists r.B \qquad\qquad \text{empty } \sqcap = \top \\ A &\sqsubseteq \forall r.B \qquad\qquad \text{empty } \sqcup = \bot \end{aligned}$$

Finite query entailment

Definition

A query Q is *finitely entailed* by a graph G and TBox $\mathcal{T}$, written

$$G, \mathcal{T} \models_{\mathsf{fin}} Q\,,$$

if Q holds in every finite graph H that contains G as a subgraph and satisfies $\mathcal{T}$.

Problem

Given a finite graph G, an $\mathcal{ALC}$ TBox $\mathcal{T}$, and a UCRPQ Q, decide if $G, \mathcal{T} \models_{\mathit{fin}} Q$.

Finiteness paradox

Does $\mathcal{T} = \{\mathsf{Person} \sqsubseteq \exists\, \mathsf{childOf}.\, \mathsf{Person}\}$ model reality well?

Finiteness paradox

Does $\mathcal{T} = \{\mathsf{Person} \sqsubseteq \exists\, \mathsf{childOf}.\, \mathsf{Person}\}$ model reality well?

Suppose we know that at least one person exists:

the input graph G contains a node with label Person.

Finiteness paradox

Does $\mathcal{T} = \{\mathsf{Person} \sqsubseteq \exists\, \mathsf{childOf}.\, \mathsf{Person}\}$ model reality well?

Suppose we know that at least one person exists:

the input graph G contains a node with label Person.

Then, over finite models we can conclude that somebody is their own ancestor...

$$G, \mathcal{T} \models_{\mathsf{fin}} \exists x\ \mathsf{childOf}^{+}(x, x)$$

Should we embrace infinite models or rethink our modelling?

Plan of Attack

1. Reduce to the case with trivial input graph.
2. Unrestricted entailment of CRPQs.
3. Finite entailment of CQs

   by reduction to unrestricted entailment of CQs.
4. Finite entailment of simple CRPQs

   by reduction to finite entailment of CQs.
5. Finite entailment of CRPQs

   by reduction to finite entailment of simple CRPQs.

Handling data: starlike graphs & factorization

Starlike countermodels

Definition

A *starlike* graph H consists of a *central part* H_0 and pairwise disjoint *peripheral parts* $H_1, \dots, H_k$ each sharing exctly one node with the central part.

Theorem

$G, \mathcal{T} \not\models Q$ iff there is a starlike graph H such that $H \models \mathcal{T}$, $H \not\models Q$, $G \subseteq H$, and moreover $H_0 = G$ (up to additional node labels in H_0) and $H_i \models \mathcal{T}$ for $i = 1, \dots, k$.

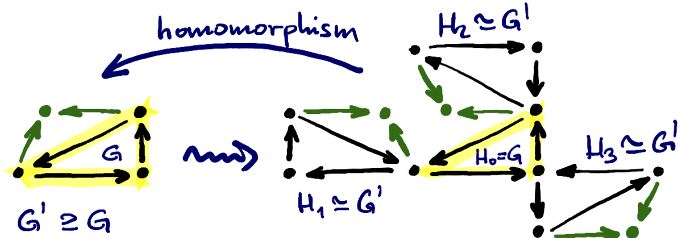
Starlike countermodels

Definition

A *starlike* graph H consists of a *central part* H_0 and pairwise disjoint *peripheral parts* $H_1, \ldots, H_k$ each sharing exctly one node with the central part.

Theorem

$G, \mathcal{T} \not\models Q$ *iff there is a starlike graph* H *such that* $H \models \mathcal{T}$, $H \not\models Q$, $G \subseteq H$, *and moreover* $H_0 = G$ *(up to additional node labels in* H_0*) and* $H_i \models \mathcal{T}$ *for* $i = 1, \ldots, k$.



Homomorphisms

Definition

A *homomorphism* $h : G \to G'$ is a function $h : \mathsf{dom}(G) \to \mathsf{dom}(G')$ such that

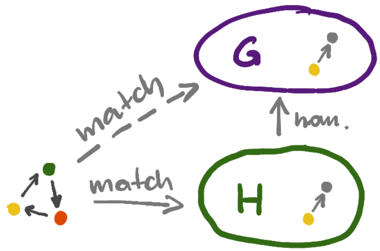
- v has label A in G iff $h(v)$ has label A in G';
- an r-edge from u to v in G implies an r-edge from $h(u)$ to $h(v)$ in G'.

Let $\mathcal{T}_\forall$ be $\mathcal{T}$ with all *participation constraints* $A \sqsubseteq \exists r.B$ dropped

Fact

If there is a homomorphism $H \to G$, then

- *if $G \models \mathcal{T}_\forall$ then $H \models \mathcal{T}_\forall$;*
- *if $G \not\models Q$ then $H \not\models Q$.*



Factorization

Theorem

For each connected UCRPQ Q *one can construct effectively a connected UCRPQ* $\widehat{Q}$ *using additional fresh node labels (and their complements!) such that*

- $\widehat{Q}$ *holds in a starlike graph iff it holds in any of its parts;*
- $\widehat{Q}$ *is almost equivalent to* Q*.*

If Q *is a UCQ or a simple UCRPQ, so is* $\widehat{Q}$*.*

Factorization

Theorem

For each connected UCRPQ Q one can construct effectively a connected UCRPQ $\widehat{Q}$ using additional fresh node labels (and their complements!) such that

- *$\widehat{Q}$ holds in a starlike graph iff it holds in any of its parts;*
- *$\widehat{Q}$ is almost equivalent to Q.*

If Q is a UCQ or a simple UCRPQ, so is $\widehat{Q}$.

Definition

$\widehat{Q}$ is *almost equivalent* to Q if for every graph G,
Q holds in G iff $\widehat{Q}$ holds in every graph $\widehat{G}$ equal to G up to $\Gamma_{\widehat{Q}} \setminus \Gamma_Q$.

Factorization

Theorem

For each connected UCRPQ Q one can construct effectively a connected UCRPQ $\widehat{Q}$ using additional fresh node labels (and their complements!) such that

- *$\widehat{Q}$ holds in a starlike graph iff it holds in any of its parts;*
- *$\widehat{Q}$ is almost equivalent to Q.*

If Q is a UCQ or a simple UCRPQ, so is $\widehat{Q}$.

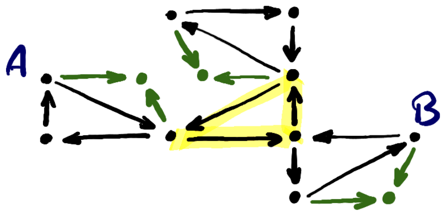
Definition

$\widehat{Q}$ is *almost equivalent* to Q if for every graph G,
Q holds in G iff $\widehat{Q}$ holds in every graph $\widehat{G}$ equal to G up to $\Gamma_{\widehat{Q}} \setminus \Gamma_Q$.

Fact

If $\widehat{Q}$ is almost equivalent to Q, and neither $\mathcal{T}$ nor G use labels from $\Gamma_{\widehat{Q}} \setminus \Gamma_Q$, then $G, \mathcal{T} \models_{\mathsf{fin}} Q$ iff $G, \mathcal{T} \models_{\mathsf{fin}} \widehat{Q}$.

Factorization: example



If $Q = A(x) \wedge x \xrightarrow{*} y \wedge B(y)$, for $\widehat{Q}$ we can take the union of

$$A(x) \wedge \bar{A}_*(x)\,, \quad A_*(x) \wedge x \xrightarrow{*} z \wedge \bar{A}_*(z)\,, \quad A_*(z) \wedge B_*(z)\,,$$
$$\bar{B}_*(z) \wedge z \xrightarrow{*} y \wedge B_*(y), \quad \bar{B}_*(y) \wedge B(y)\,.$$

$\widehat{Q}$ detects if label A_* is missing in a node reachable from A or label B_* is missing in a node from which B is reachable, or some node has both label A_* and label B_*.
If labels A_* and B_* are placed correctly in $\widehat{H}$, then $\widehat{H} \models \widehat{Q}$ iff $H \models Q$; if not, $\widehat{H} \models \widehat{Q}$.
Place A_* and B_* arbitrarily in $\widehat{H}$. Suppose e.g. $\widehat{H} \models A_*(x) \wedge x \xrightarrow{*} z \wedge \bar{A}_*(z)$.
Why must some part of $\widehat{H}$ satisfy this disjunct?

Eliminating data

Definition

A *type* τ is a consistent set of node labels and complement node labels.

$\tau, \mathcal{T} \models_{\text{fin}} \widehat{Q}$ iff there is no finite G s.t. $G \models \mathcal{T}$, $G \not\models Q$, and G has a node of type τ.

Fact

$G, \mathcal{T} \not\models_{\text{fin}} Q$ iff for some $\widehat{G}$ obtained from G by adding labels, $\widehat{G} \not\models \widehat{Q}$, $\widehat{G} \models \mathcal{T}_\forall$, and $\tau, \mathcal{T} \not\models_{\text{fin}} \widehat{Q}$ for each type τ over $\Gamma_\mathcal{T} \cup \Gamma_{\widehat{Q}}$ realized in $\widehat{G}$.

Proof.

- $G, \mathcal{T} \not\models_{\text{fin}} Q$ iff $G, \mathcal{T} \not\models_{\text{fin}} \widehat{Q}$.
- Equivalently, there is a starlike graph H s.t. $G \subseteq H$, $H \models \mathcal{T}$, $H \not\models \widehat{Q}$, and
  - $H_0 = G$ (up to additonal labels in H_0),
  - $H_i \models \mathcal{T}$ for $i = 1, \ldots, k$.
- Equivalently, there is a starlike graph H s.t.
  - $H_0 = G$ (up to additonal labels in H_0), $H_0 \models \mathcal{T}_\forall$, $H_0 \not\models \widehat{Q}$;
  - $H_i \models \mathcal{T}$ and $H_i \not\models \widehat{Q}$ for $i = 1, \ldots, k$.

□

We have seen that

- in order to solve $G, \mathcal{T} \models Q$ it suffices to solve $\tau, \mathcal{T} \models Q$, and similarly for $\models_{\mathsf{fin}}$.

We used

- starlike graphs, homomorphisms, query factorization.

Next, we

- solve $\tau, \mathcal{T} \models Q$.

The simplicity of infinity: unravelling & fixpoints

Focus problem

Problem

Given an $\mathcal{ALC}$ TBox $\mathcal{T}$, a factorized UCRPQ $\widehat{Q}$, and a type τ over $\Gamma_{\mathcal{T}} \cup \Gamma_{\widehat{Q}}$, decide if $\tau, \mathcal{T} \models \widehat{Q}$.

Strategy

- Show a simple-countermodel property.
- Show how to find simple countermodels.

Countermodels with bounded out-degree

Lemma

$\tau, \mathcal{T} \not\models Q$ iff some graph of out-degree at most $|\mathcal{T}|$ is a countermodel.

Proof.

- Take any countermodel H.
- For each participation constraint $A \sqsubseteq \exists r.B$ in $\mathcal{T}$ and each node v with label A, pick an r-edge from v to a node with label B, and paint it red.
- Let H' be obtained from H by all nodes but only red edges.
- H' has out-degree at most $|\mathcal{T}|$ and is a countermodel:
  - $H' \to H$, so $H' \models \mathcal{T}_\forall$ and $H' \not\models Q$;
  - H' contains all nodes of H, so it has a node of type τ;
  - by construction, it satisfies all participation constraints in $\mathcal{T}$.

$\square$

Tree countermodel property

Theorem (tree countermodel property)

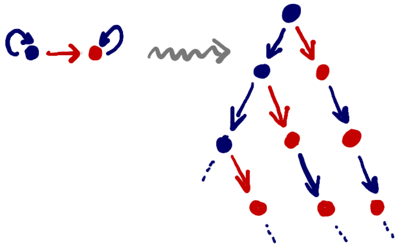
$\tau, \mathcal{T} \not\models Q$ *iff some* $|\mathcal{T}|$*-ary tree is a countermodel.*

Unravelling

Definition

Given a graph G and node v_0, the *unravelling of* G from v_0 is a tree $G(v_0)$ such that

- nodes of $G(v_0)$ are directed paths in G (also non-simple) that start in v_0;
- node $v_0 \xrightarrow{r_1} v_1 \xrightarrow{r_2} \cdots \xrightarrow{r_{k-1}} v_{k-1} \xrightarrow{r_k} v_k$ has labels inherited from v_k and an incoming r_k-edge from $v_0 \xrightarrow{r_1} v_1 \xrightarrow{r_2} \cdots \xrightarrow{r_{k-1}} v_{k-1}$.



Unravelling

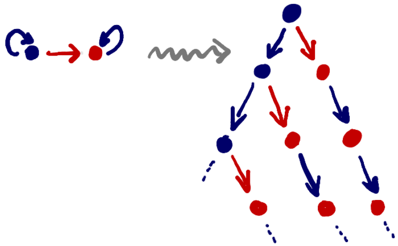
Definition

Given a graph G and node v_0, the *unravelling of* G from v_0 is a tree $G(v_0)$ such that

- nodes of $G(v_0)$ are directed paths in G (also non-simple) that start in v_0;
- node $v_0 \xrightarrow{r_1} v_1 \xrightarrow{r_2} \cdots \xrightarrow{r_{k-1}} v_{k-1} \xrightarrow{r_k} v_k$ has labels inherited from v_k and an incoming r_k-edge from $v_0 \xrightarrow{r_1} v_1 \xrightarrow{r_2} \cdots \xrightarrow{r_{k-1}} v_{k-1}$.

Fact

Let $\eta(v_0 \xrightarrow{r_1} v_1 \xrightarrow{r_2} \cdots \xrightarrow{r_k} v_k) = v_k$.
Then η *is a homomorphism from* $G(v_0)$ *to* G.



Unravelling

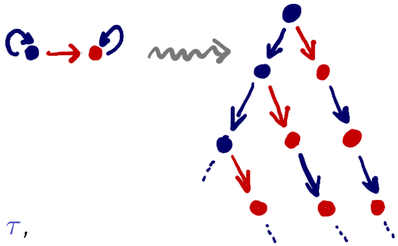
Definition

Given a graph G and node v_0, the *unravelling of* G from v_0 is a tree $G(v_0)$ such that

- nodes of $G(v_0)$ are directed paths in G (also non-simple) that start in v_0;
- node $v_0 \xrightarrow{r_1} v_1 \xrightarrow{r_2} \cdots \xrightarrow{r_{k-1}} v_{k-1} \xrightarrow{r_k} v_k$ has labels inherited from v_k and an incoming r_k-edge from $v_0 \xrightarrow{r_1} v_1 \xrightarrow{r_2} \cdots \xrightarrow{r_{k-1}} v_{k-1}$.

Fact

Let $\eta(v_0 \xrightarrow{r_1} v_1 \xrightarrow{r_2} \cdots \xrightarrow{r_k} v_k) = v_k$.
Then η *is a homomorphism from* $G(v_0)$ *to* G.



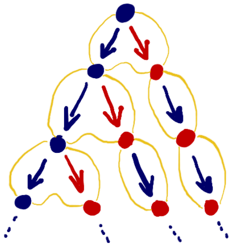
Corollary

If H *is a countermodel for* $\tau, \mathcal{T} \models Q$ *and* v_0 *has type* τ,
then $H(v_0)$ *is a countermodel, too.*

Finding tree-shaped countermodels

A tree is a countermodel for $\tau, \mathcal{T} \models Q$
iff some tile's root has type τ and each tile:

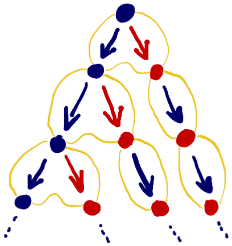
- satisfies $\mathcal{T}_\forall$;
- does not satisfy $\widehat{Q}$ (why?);
- satisfies participation constraints for the tile's root.



Finding tree-shaped countermodels

A tree is a countermodel for $\tau, \mathcal{T} \models Q$
iff some tile's root has type τ and each tile:

- satisfies $\mathcal{T}_\forall$;
- does not satisfy $\widehat{Q}$ (why?);
- satisfies participation constraints for the tile's root.



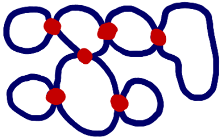
Greatest fixpoint algorithm for testing existence of a tree-shaped countermodel:

1. Compute the set of legal tiles (of degree bounded by $|\mathcal{T}|$).
2. Iteratively remove tiles that contain a child for which there is no tile with matching root, until the set stabilizes.
3. ACCEPT iff the resulting set contains a tile with root of type τ.

Factorization revisited

Lemma

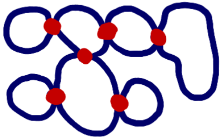
Query $\widehat{Q}$ holds in a bubbletree graph iff it holds in any of its parts.

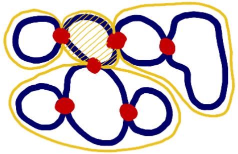


Factorization revisited

Lemma

Query $\widehat{Q}$ holds in a bubbletree graph iff it holds in any of its parts.





Proof.

1. If a bubbletree H satisfies $\widehat{Q}$, some finite sub-bubbletree H' satisfies $\widehat{Q}$.
2. Every bubbletree is a starlike graph: any bubble can be the central part, the remaining connected sub-bubbletrees are then peripheral parts.
3. If a finite bubbletree with at least two bubbles satisfies $\widehat{Q}$, some strictly smaller sub-bubbletree satisfies $\widehat{Q}$.
4. Applying 3 multiple times we arrive at a single bubble that satisfies $\widehat{Q}$.

□

We have seen that

- UCRPQs have a tree coutermodel property (with respect to $\mathcal{ALC}$);
- $\tau, \mathcal{T} \models Q$ can be solved for UCRPQs.

We used

- unravellings, query factorization (again!), TBox factorization
- type elimination (greatest fixpoint algorithm)

Next, we

- solve $\tau, \mathcal{T} \models_{\text{fin}} Q$ for CQs.

Into the finite: coloured blocking

Focus problem

Problem

Given a type τ, an $\mathcal{ALC}$ TBox $\mathcal{T}$, and a conjunctive query Q, decide if $\tau, \mathcal{T} \models_{\mathit{fin}} Q$.

Focus problem

Problem

Given a type τ, an $\mathcal{ALC}$ TBox $\mathcal{T}$, and a conjunctive query Q, decide if $\tau, \mathcal{T} \models_{\mathit{fin}} Q$.

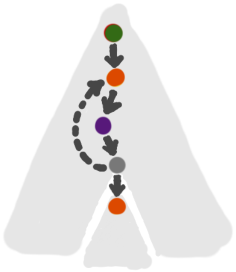
Theorem (finite countermodel property AKA finite controllability)

For every type τ, $\mathcal{ALC}$ TBox $\mathcal{T}$, and conjunctive query Q,

$$\tau, \mathcal{T} \models Q \quad \textit{if and only if} \quad \tau, \mathcal{T} \models_{\mathit{fin}} Q.$$

- $\models$ always implies $\models_{\text{fin}}$: if something holds for all models, then in particular it holds for all finite models.
- For the converse implication we assume that $\tau, \mathcal{T} \not\models Q$, take the (infinite) tree countermodel and turn it into a finite countermodel, as follows.

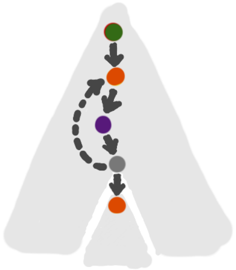
Finite countermodel property for CQs (finite controllability)



Blocking

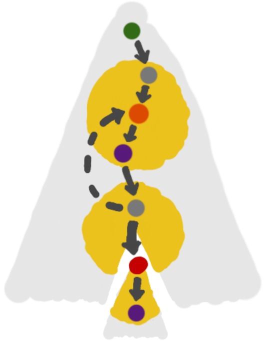
Loop back whenever type repeats.
Works for finite *model* property. For finite controllability short cycles must be avoided.

Finite countermodel property for CQs (finite controllability)



Blocking

Loop back whenever type repeats.
Works for finite *model* property. For finite controllability short cycles must be avoided.



Coloured blocking

First, colour all elements so that no colour repeats within any neighbourhood of radius $|Q|$. Loop back whenever the whole coloured neighbourhood repeats.

Colouring

Lemma

Fix a radius n. A graph G of bounded degree can be coloured with finitely many colours such that no colour repeats within any neighbourhood of radius n.

Proof.

- Because degree is bounded, a neighbourhood of radius $2n$ has size bounded by some m. We shall use m colours.
- Colour nodes greedily *ad infinitum*: if a node has no colour yet, at most $m-1$ colours are used within radius $2n$, so pick one that is still free.
- Consider two nodes in some neighbourhood of radius n. Then, the one coloured later saw the colour of the other one within radius $2n$, and did not use it.

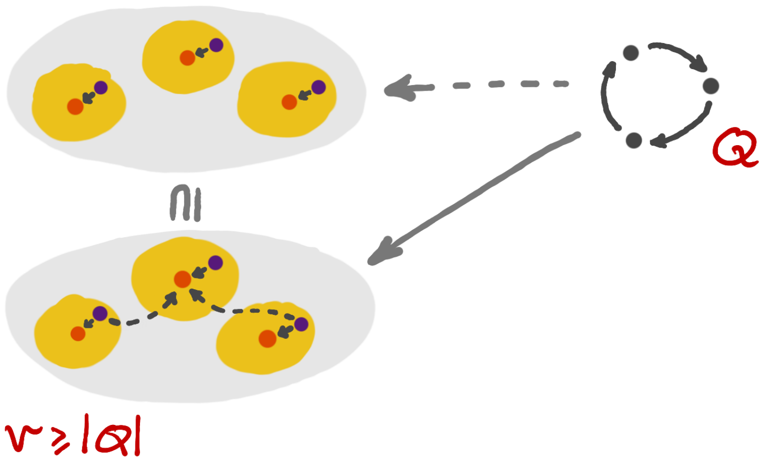
□

Redirecting edges

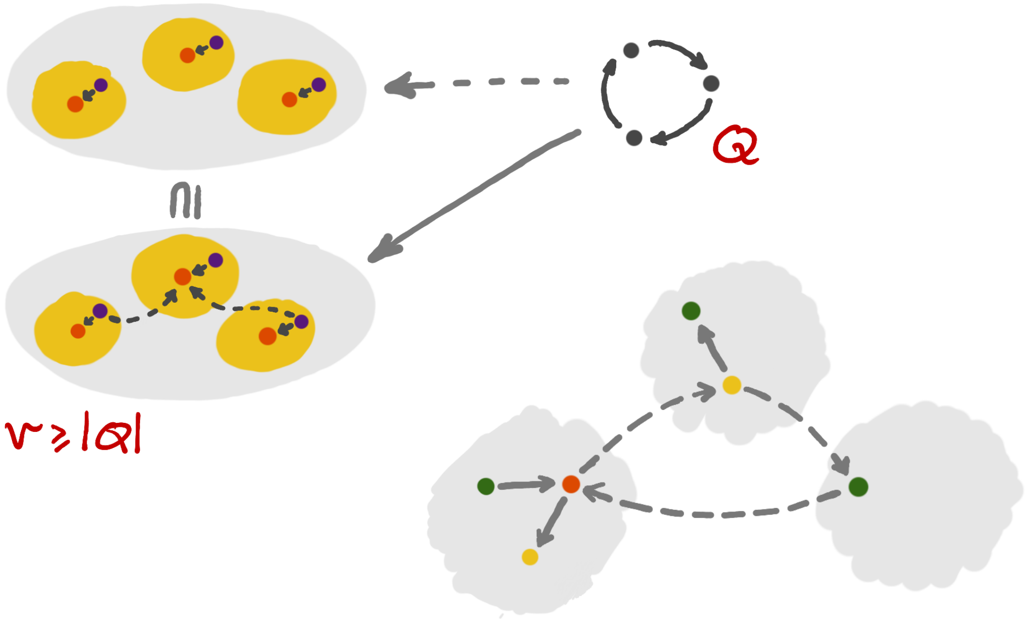
Theorem

Let G be a graph such that in each neighbourhood of radius n, all nodes have different colours. Let G' be obtained from G by redirecting some edges such that the old target and the new target have isomorphic neighbourhoods of radius n in G. Then, for each CQ Q with at most n binary atoms, if $G \not\models Q$, then $G' \not\models Q$.

Proof by example



Proof by example



We have seen that

- CQs have a finite coutermodel property (with respect to $\mathcal{ALC}$);
- $\tau, \mathcal{T} \models_{\mathsf{fin}} Q$ can be solved for CQs

We used

- coloured blocking.

Next, we

- solve $\tau, \mathcal{T} \models_{\mathsf{fin}} Q$ for simple UCRPQs.

Beyond conjunctive queries: connectivity & abstraction

Focus problem

Problem

Given a type τ, an $\mathcal{ALC}$ TBox $\mathcal{T}$, and a simple UCRPQ Q, decide if $\tau, \mathcal{T} \models_{\mathit{fin}} Q$.

Simple UCRPQ only use regular expressions of the form r and $(r_1 + r_2 + \cdots + r_k)^*$.

In fact, we will first solve the problem for the role-blind case:

- $r_1, \ldots, r_k$ above are all roles in $\Sigma_{\mathcal{T}}$ (the set of roles used in $\mathcal{T}$).

No finite countermodel property

Query Q detects a cycle connecting labels A and B:

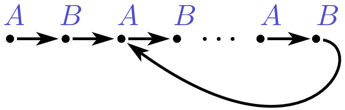
$$\exists x\, \exists y\, A(x) \wedge B(y) \wedge r^*(x,y) \wedge r^*(y,x)$$

For type $\tau = \{A\}$ and TBox $\mathcal{T}$ containing

$$\top \sqsubseteq \exists r.\top$$

$$A \sqsubseteq \forall r.\neg A \sqcap B$$

$$B \sqsubseteq \forall r.\neg B \sqcap A$$

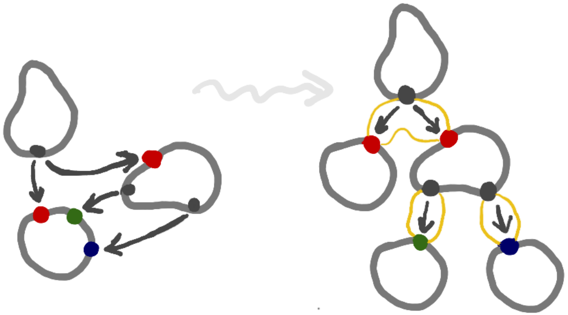


there is a countermodel, but not a finite one.

Simple countermodel property

Lemma

In the role-blind case, $\tau, \mathcal{T} \not\models_{\mathit{fin}} Q$ iff there is a finite bubbletree countermodel with alternating small and large bubbles, such that all large bubbles are strongly connected.



Simple countermodel property – refined

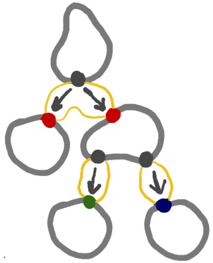
Lemma

In the role-blind case, $\tau, \mathcal{T} \not\models_{\mathit{fin}} Q$ iff there is a finite bubbletree with alternating small and large bubbles such that some bubble's root has type τ, each large bubble satisfies

- *satisfies* $\widehat{\mathcal{T}} = \mathcal{T}_\forall \cup \{A \sqsubseteq \mathsf{Need}_{r,B} \sqcup \exists r.B \mid A \sqsubseteq \exists r.B \in \mathcal{T}\}$,
- *does not satisfy* $\widehat{Q} \bmod \Sigma^*_\mathcal{T}$, *defined as* $\widehat{Q}$ *without* $\Sigma^*_\mathcal{T}$ *atoms,*

and each small bubble (tile)

- *satisfies* $\widetilde{\mathcal{T}} = \mathcal{T}_\forall \cup \{\mathsf{Need}_{r,B} \sqsubseteq \exists r.B \mid A \sqsubseteq \exists r.B \in \mathcal{T}\}$,
- *does not satisfy* $\widehat{Q}$,



Proof.

If such a bubbletree exists, it satisfies $\mathcal{T}$ and has a node of type τ.
Also, no bubble satisfies $\widehat{Q}$, so the whole bubbletree does not satisfy $\widehat{Q}$.
If a finite bubbletree with strongly connected large bubbles is a countermodel for $\widehat{Q}$, then no bubble satisfies $\widehat{Q}$ and conditions involving $\widehat{\mathcal{T}}$, $\widetilde{\mathcal{T}}$, and τ hold when labels $\mathsf{Need}_{r,B}$ are placed suitably. But then, no large bubble satisfies $\widehat{Q} \bmod \Sigma^*_\mathcal{T}$. $\square$

Finding simple countermodels

If we had the set of legal large bubbless, we could use a least fixpoint algorithm to test if a suitable finite bubbletree exists:

1. Start from the set of legal tiles that have no children (no witnesses needed).
2. Iteratively add
   - large bubbles, whenever all their nodes have a tile with matching root in the set,
   - tiles, whenever all their children have a large bubble with matching root in the set,

   until the set stabilizes.
3. Accept if the resulting set contains a bubble with root of type τ.

But there are infinitely many large bubbles! How do we get the legal ones?

Finding legal large bubbles

Observation 1

The fixpoint algorithm only cares about

- the type of the root,
- possible types of nodes.

Indeed, for nodes of the same type we can use the same tiles.

Observation 2

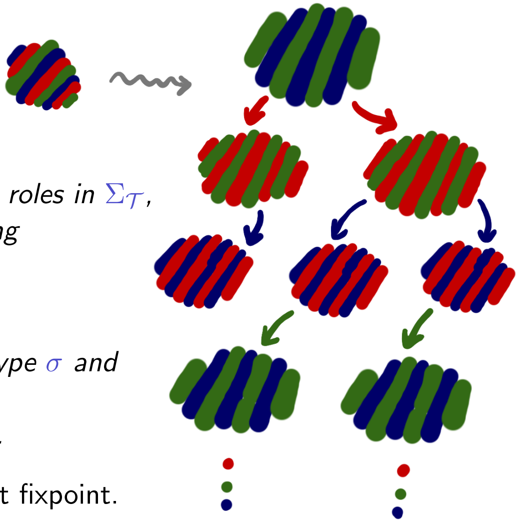
A large bubble with root of type σ and nodes of types from Θ exists iff

$$\sigma, \widehat{\mathcal{T}} \cup \mathcal{T}_\Theta \not\models_{\mathsf{fin}} \widehat{Q} \bmod \Sigma_{\mathcal{T}}^* \quad \text{where} \quad \mathcal{T}_\Theta = \left\{ \top \sqsubseteq \bigsqcup_{\theta \in \Theta} \bigsqcap_{A \in \theta} A \right\}$$

.

Role-aware case

$\widehat{Q} \bmod \Sigma_{\mathcal{T}}^*$ is not a CQ, but none of its reachability atoms can traverse all roles.



Lemma

If no reachability atom in Q can traverse all roles in $\Sigma_{\mathcal{T}}$, then $\tau, \mathcal{T} \not\models_{\mathit{fin}} Q$ iff there is a role-alternating bubbletree countermodel.

Lemma

There is a legal r-free bubble with root of type σ and nodes of types from Θ iff $\sigma, \mathcal{T} - r \not\models_{\mathit{fin}} \widehat{Q}$, where $\mathcal{T} - r$ is $\mathcal{T}$ without CIs mentioning r.

- Bubbletree may be infinite: use greatest fixpoint.
- We can bound the size of bubbles: pick one for each r, σ, and Θ.
- Reachability atoms traverse less than $|\Sigma_{\mathcal{T}}|$ tiles, so $\widehat{Q}$ behaves like a CQ.
- Use coloured blocking at the level of bubbles to get a finite countermodel.

We have seen that

- simple UCRPQs do not have a finite coutermodel property;
- $\tau, \mathcal{T} \models_{\mathsf{fin}} Q$ can be solved for simple UCRPQs.

We used

- decompositions into strongly connected components,
- abstract representation of subgraphs (bubbles)
- type elimination (again!) and type “construction” (least fixpoint algorithm)

Next, we

- solve $\tau, \mathcal{T} \models_{\mathsf{fin}} Q$ for general UCRPQs.

Regular reachability: tape construction

Focus problem

Problem

Given a type τ, an $\mathcal{ALC}$ TBox $\mathcal{T}$, and a UCRPQ Q, decide if $\tau, \mathcal{T} \models_{\mathit{fin}} Q$.

Focus problem

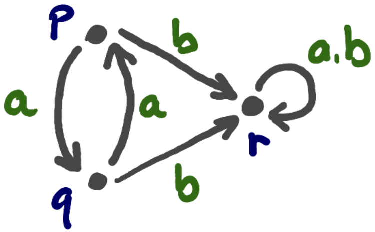
Problem

Given a type τ, an $\mathcal{ALC}$ TBox $\mathcal{T}$, and a UCRPQ Q, decide if $\tau, \mathcal{T} \models_{\mathit{fin}} Q$.

Theorem

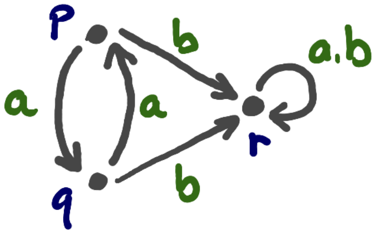
Given a type τ, an $\mathcal{ALC}$ TBox $\mathcal{T}$, and a UCRPQ Q, one can compute an $\mathcal{ALC}$ TBox $\mathcal{T}'$ and a simple UCRPQ Q' such that $\tau, \mathcal{T} \models_{\mathit{fin}} Q$ iff $\tau, \mathcal{T}' \models_{\mathit{fin}} Q'$.

UCRPQs via automata



- Represent the UCRPQ using a common deterministic automaton.
- Instead of regular expressions use pairs of states of the automaton: $\langle p, q\rangle(x, y)$, meaning *there is a path from* x *to* y *labelled with a word* w *such that* $p \xrightarrow{w} q$
- For example, $A(x) \wedge (aa)^*(x, y) \wedge B(y) \wedge b(y, z) \wedge C(z) \wedge (a + b)^*(z, x)$ can be written as

UCRPQs via automata



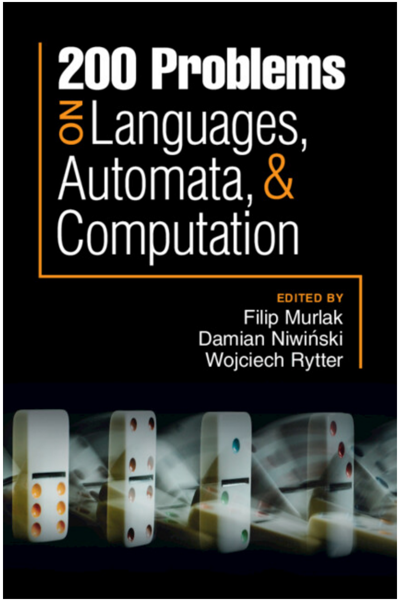
- Represent the UCRPQ using a common deterministic automaton.
- Instead of regular expressions use pairs of states of the automaton: $\langle p, q\rangle(x, y)$, meaning *there is a path from* x *to* y *labelled with a word* w *such that* $p \xrightarrow{w} q$
- For example, $A(x) \wedge (aa)^*(x, y) \wedge B(y) \wedge b(y, z) \wedge C(z) \wedge (a + b)^*(z, x)$ can be written as $A(x) \wedge \langle p, p\rangle(x, y) \wedge B(y) \wedge \langle p, r\rangle(y, z) \wedge C(z) \wedge \langle r, r\rangle(z, x)$

Toy Problem

Fix a deterministic finite automaton.
Given a word w, build a structure
of size $O(|w|)$ that allows answering
queries of the form $p \xrightarrow{w[i,j]} q$
in time $O(1)$.

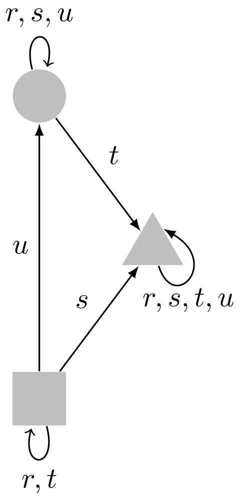
Toy Problem

Fix a deterministic finite automaton.
Given a word w, build a structure of size $O(|w|)$ that allows answering queries of the form $p \xrightarrow{w[i,j]} q$ in time $O(1)$.



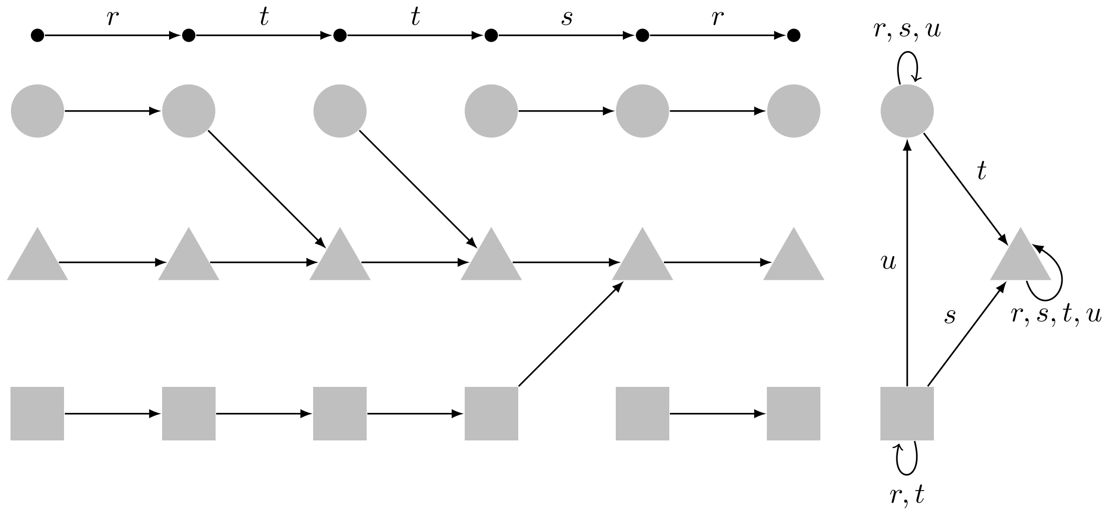
Tape construction

$$\begin{tikzcd}
\bullet \arrow[r, "r"] & \bullet \arrow[r, "t"] & \bullet \arrow[r, "t"] & \bullet \arrow[r, "s"] & \bullet \arrow[r, "r"] & \bullet
\end{tikzcd}$$



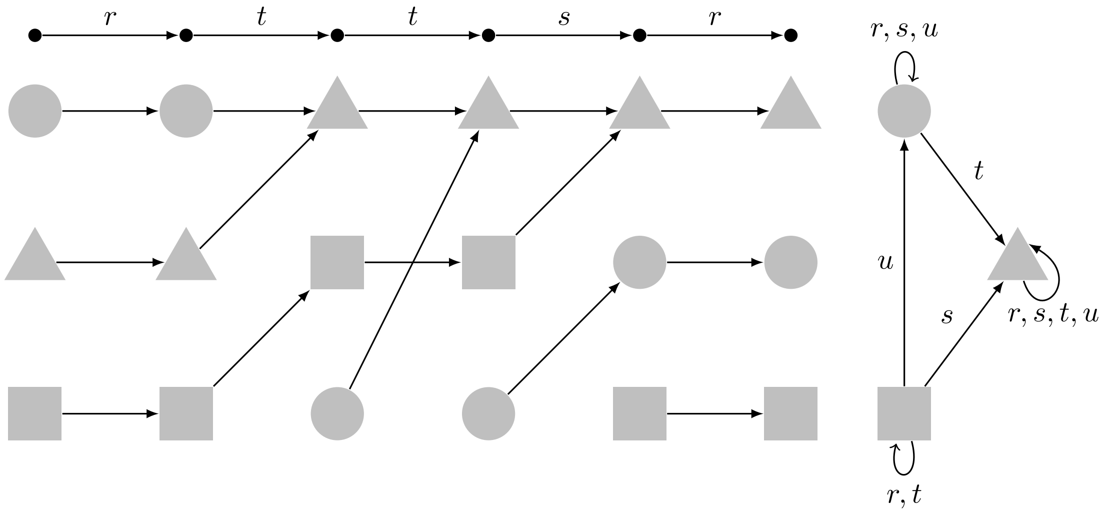
[Hesse 2003; Bojańczyk 2009; Gutowski 2022]

Tape construction



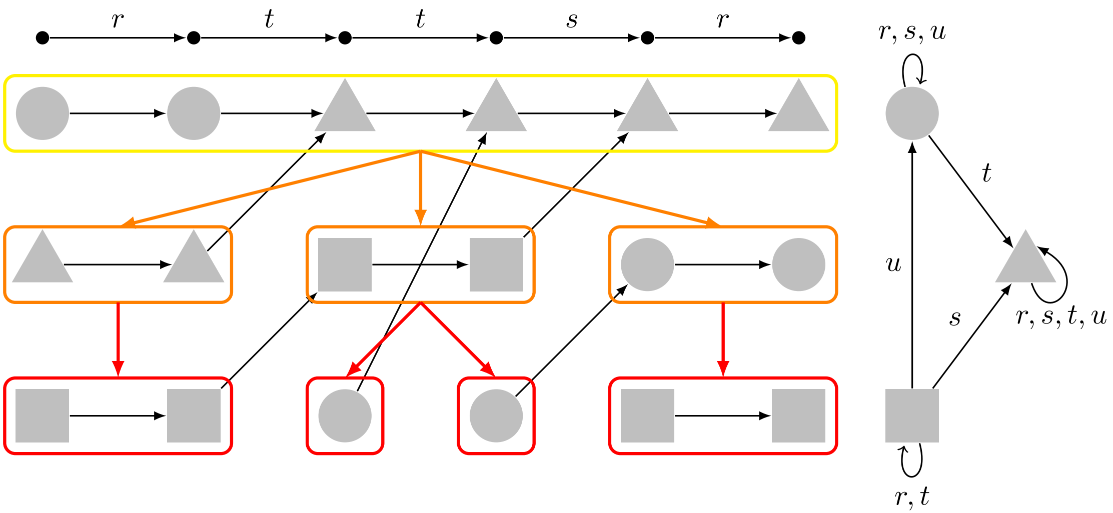
[Hesse 2003; Bojańczyk 2009; Gutowski 2022]

Tape construction



[Hesse 2003; Bojańczyk 2009; Gutowski 2022]

Tape construction



[Hesse 2003; Bojańczyk 2009; Gutowski 2022]

Construction of the new TBox

Work with graphs consistently labelled with macrostates and macrotransitions.

Construction of the new TBox

Work with graphs consistently labelled with macrostates and macrotransitions.

- Decorate nodes with macrostates $\mathbf{p}$ (state permutations):

$$\top \sqsubseteq \bigsqcup_{\mathbf{p}} \mathbf{p} \qquad \bigsqcup_{\mathbf{p} \neq \mathbf{p}} \mathbf{p} \sqcap \mathbf{q} \sqsubseteq \bot$$

Construction of the new TBox

Work with graphs consistently labelled with macrostates and macrotransitions.

- Decorate nodes with macrostates $\mathbf{p}$ (state permutations):

$$\top \sqsubseteq \bigsqcup_{\mathbf{p}} \mathbf{p} \qquad \bigsqcup_{\mathbf{p} \neq \mathbf{p}} \mathbf{p} \sqcap \mathbf{q} \sqsubseteq \bot$$

- Replace roles r with macrotransitions $\langle \mathbf{p}, r, \mathbf{q} \rangle$ (transitions between macrostates)

$$A \sqsubseteq \bigsqcap_{\langle \mathbf{p}, r, \mathbf{q} \rangle} \forall \langle \mathbf{p}, r, \mathbf{q} \rangle . B \qquad A' \sqsubseteq \bigsqcup_{\langle \mathbf{p}, r', \mathbf{q} \rangle} \exists \langle \mathbf{p}, r', \mathbf{q} \rangle . B'$$

for all $A \sqsubseteq \forall r.B$ and $A' \sqsubseteq \exists r'.B'$ in $\mathcal{T}$.

Construction of the new TBox

Work with graphs consistently labelled with macrostates and macrotransitions.

- Decorate nodes with macrostates $\mathbf{p}$ (state permutations):

$$\top \sqsubseteq \bigsqcup_{\mathbf{p}} \mathbf{p} \qquad \bigsqcup_{\mathbf{p} \neq \mathbf{p}} \mathbf{p} \sqcap \mathbf{q} \sqsubseteq \bot$$

- Replace roles r with macrotransitions $\langle \mathbf{p}, r, \mathbf{q} \rangle$ (transitions between macrostates)

$$A \sqsubseteq \bigsqcap_{\langle \mathbf{p}, r, \mathbf{q} \rangle} \forall \langle \mathbf{p}, r, \mathbf{q} \rangle . B \qquad A' \sqsubseteq \bigsqcup_{\langle \mathbf{p}, r', \mathbf{q} \rangle} \exists \langle \mathbf{p}, r', \mathbf{q} \rangle . B'$$

  for all $A \sqsubseteq \forall r.B$ and $A' \sqsubseteq \exists r'.B'$ in $\mathcal{T}$.

- Ensure that $\langle \mathbf{p}, r, \mathbf{q} \rangle$-edges go from $\mathbf{p}$-nodes to $\mathbf{q}$-nodes:

$$\top \sqsubseteq \forall \langle \mathbf{p}, r, \mathbf{q} \rangle . \mathbf{q} \qquad \exists \langle \mathbf{p}, r, \mathbf{q} \rangle . \top \sqsubseteq \mathbf{p}$$

Construction of the simple UCRPQ

Definition

Macrotransition $\langle \mathbf{p}, r, \mathbf{q} \rangle$ *preserves level* ℓ if for each transition $p \xrightarrow{r} q$,

if p is at level ℓ in $\mathbf{p}$, then q is at level ℓ in $\mathbf{q}$.

Construction of the simple UCRPQ

Definition

Macrotransition $\langle \mathbf{p}, r, \mathbf{q} \rangle$ *preserves level* ℓ if for each transition $p \xrightarrow{r} q$,

$$\text{if } p \text{ is at level } \ell \text{ in } \mathbf{p}, \text{ then } q \text{ is at level } \ell \text{ in } \mathbf{q}.$$

Lemma

Suppose node x has state p at level ℓ, and y has state q at level ℓ. Then, a path from x to y yields word w st. $p \xrightarrow{w} q$ iff it sees only macrotransitions preserving level ℓ.

Construction of the simple UCRPQ

Definition

Macrotransition $\langle \mathbf{p}, r, \mathbf{q} \rangle$ *preserves level* ℓ if for each transition $p \xrightarrow{r} q$,

$$\text{if } p \text{ is at level } \ell \text{ in } \mathbf{p}\text{, then } q \text{ is at level } \ell \text{ in } \mathbf{q}.$$

Lemma

Suppose node x has state p at level ℓ, and y has state q at level ℓ. Then, a path from x to y yields word w st. $p \xrightarrow{w} q$ iff it sees only macrotransitions preserving level ℓ.

Construction

- Chop each RPQ into same-level segments, interleaved with single-edge atoms that increase the level (consider all possible choices of intermediate macrostates).
- Replace each segment with a simple RPQ: reachability over macrotransitions preserving level ℓ.

We have seen that

- $\tau, \mathcal{T} \models_{\mathsf{fin}} Q$ can be solved for UCRPQs.

We used

- finite automata, tape construction

Next, we

- behold the big picture.

The Big Picture

A zoo of description logics

Features can be added to $\mathcal{ALC}$, giving logics like $\mathcal{ALCHOIQ}$ or $\mathcal{SOQ}$ ($\mathcal{S}$ is shorthand for $\mathcal{ALCS}$).

$\mathcal{O}$: use constants as singleton concepts $\{a\}$	$A \sqsubseteq \exists r.\{a\}$
$\mathcal{I}$: use inverse roles r^- anywhere	$A \sqsubseteq \exists r^-.B$
$\mathcal{F}$: declare role r to be a partial function	$\mathsf{fun}(r)$
$\mathcal{Q}$: use counting quantifiers $\exists^{\leq n}$, $\exists^{\geq n}$	$A \sqsubseteq \exists^{\leq 5} r.B$
$\mathcal{S}$: declare role r to be transitive	$\mathsf{tra}(r)$
$\mathcal{H}$: declare role r to be contained in role s	$r \sqsubseteq s$

Landscape of finite entailment: conjunctive queries

Finite entailment of CQs is decidable for

- $\mathcal{GC}^2$ [Pratt-Hartmann 2009]
- $\mathcal{GF}$ [Bárány, Gottlob, Otto 2014]
- $\mathcal{SOI}$ and $\mathcal{SIF}$ [Gogacz et al. 2018]
- $\mathcal{SHI}$ [Danielski, Kieroński 2018]
- $\mathcal{SOQ}$ [Gogacz et al. 2019]

undecidable for

- $\mathcal{SHOIF}$ [Rudolph 2016]

a challenge for

- $\mathcal{ALCOIF}$

Landscape of finite entailment: UCRPQs and UC2RPQs

Finite entailment of UCRPQ is decidable for

- $\mathcal{ALCI}$ and $\mathcal{ALCQ}$ [Gutiérrez-Basulto et al. 2023 (to appear)]
  - should extend to $\mathcal{ALCOI}$ and $\mathcal{ALCOQ}$

Finite entailment of UC2RPQs (two-way UCRPQs) is undecidable for

- $\mathcal{ALCIOF}$ [Rudolph 2016]

a challenge for

- $\mathcal{ALC}$

Takeaway

Sometimes infinity is an oversimplification