Study of New Semantics for Regular Path Queries

Sara KHICHANE[1]

Paris Cité University
UFR of Computer Science

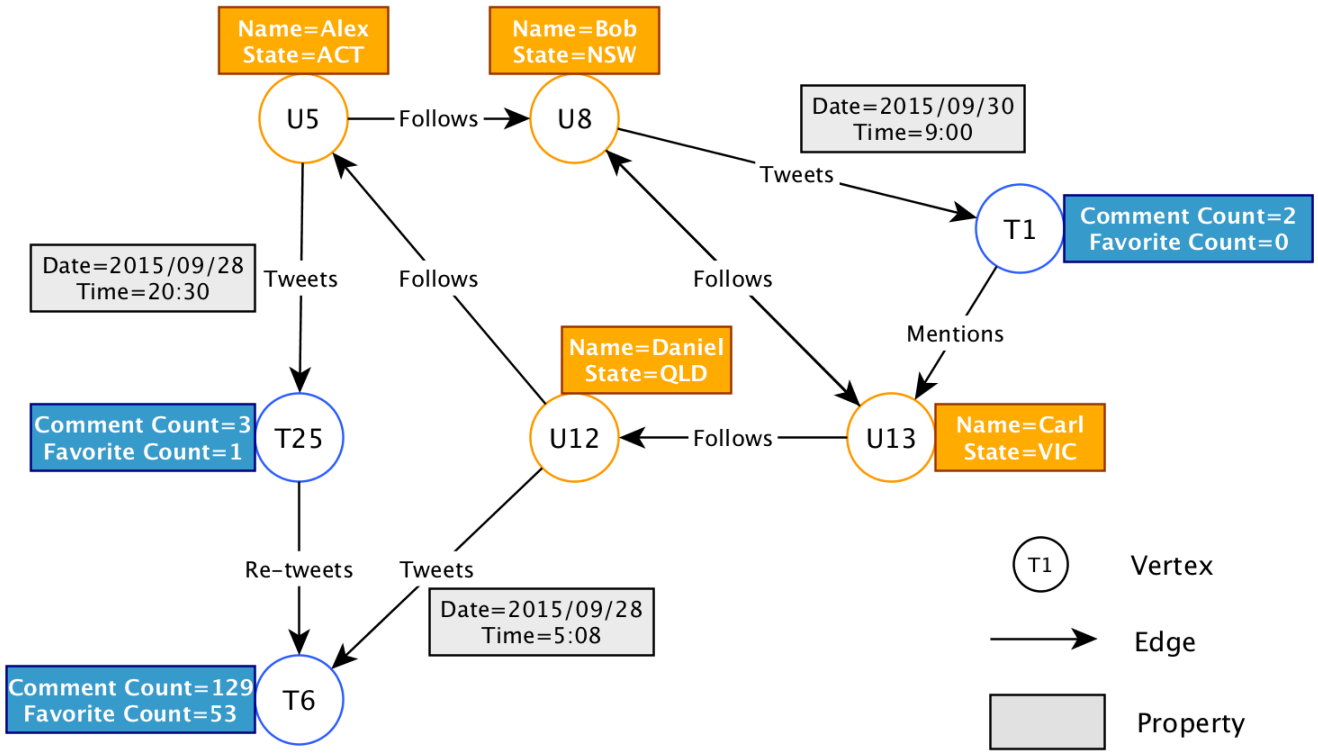
July 01, 2024

[1]Supervised by Amélie Gheerbrant, Victor Marsault and Antoine Meyer.

Introduction

Graph Database

In practice, we have property graphs.

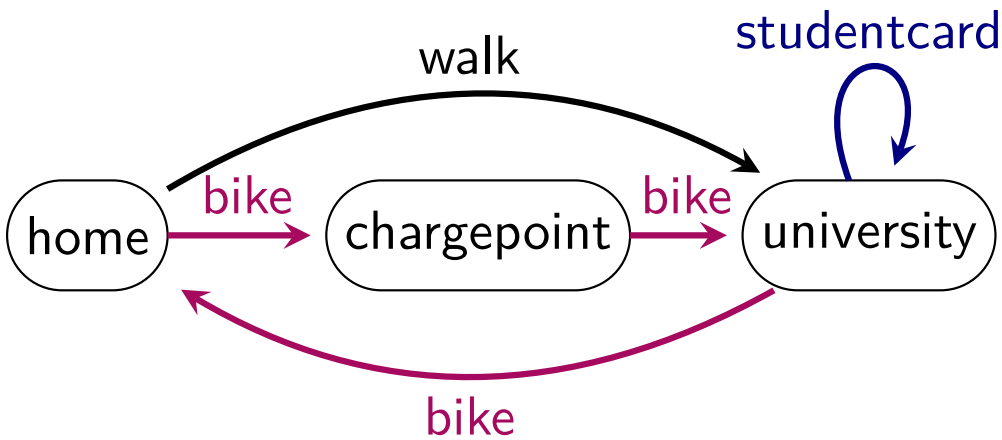


Graph Database

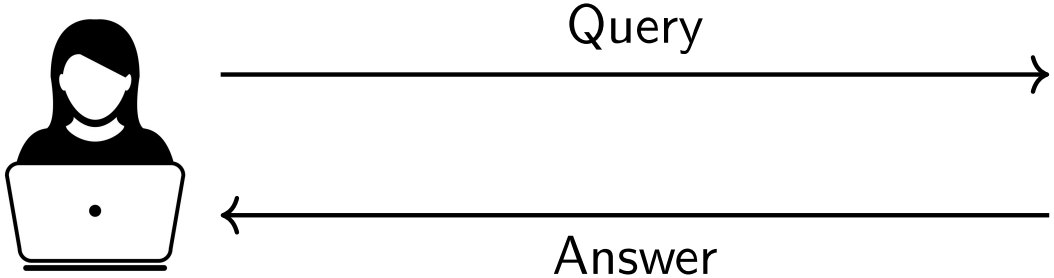
In practice, we have property graphs.

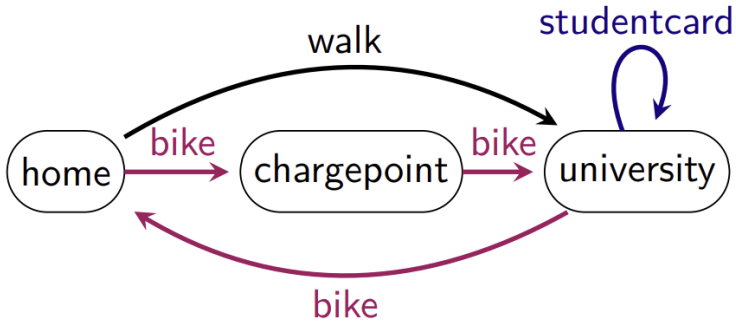
Our abstraction

Directed, multi-labeled, multi-edge graph.

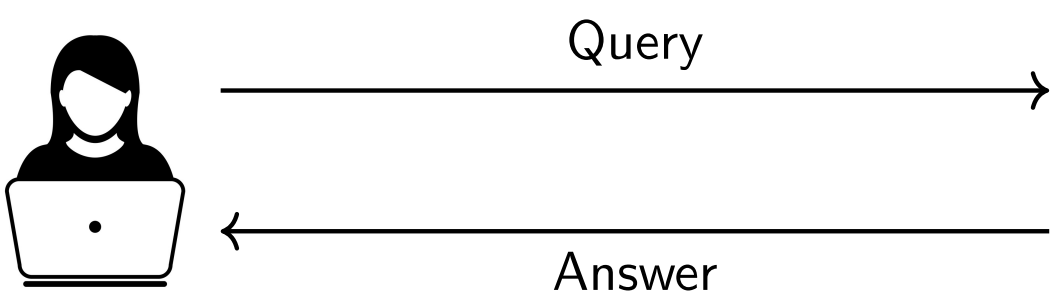


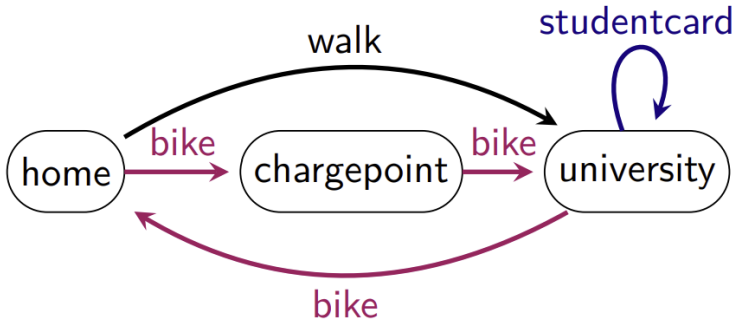
Regular Path Queries





Regular Path Queries

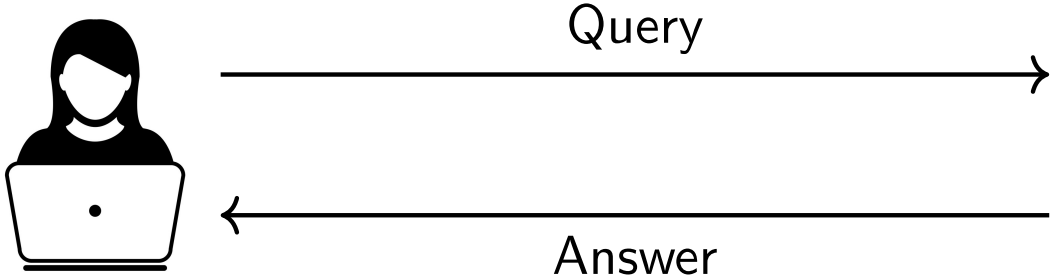


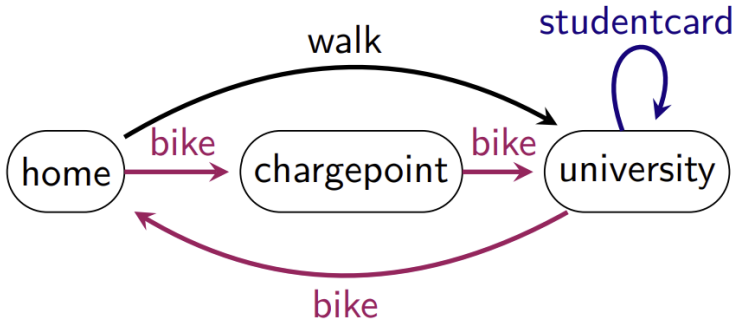


Regular Path Query

Represented by a regular expression R or an automaton A.

Regular Path Queries



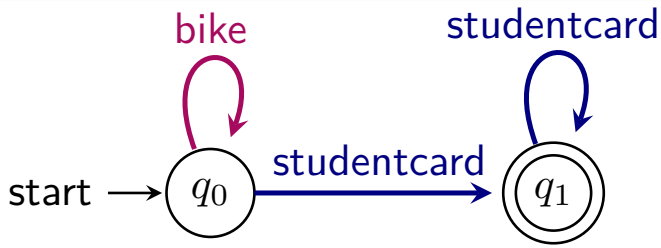


Regular Path Query

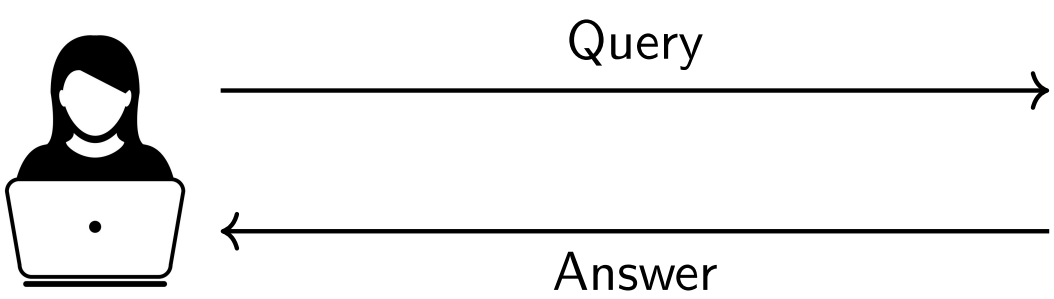
Represented by a regular expression R or an automaton A.

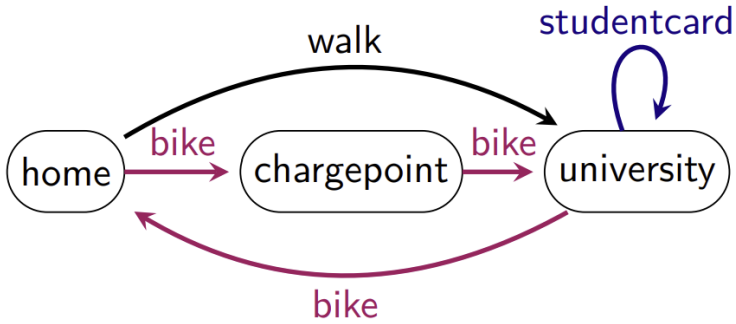
Query: From home to university

$$R = \text{bike}^* \cdot \text{studentcard}^+$$



Regular Path Queries



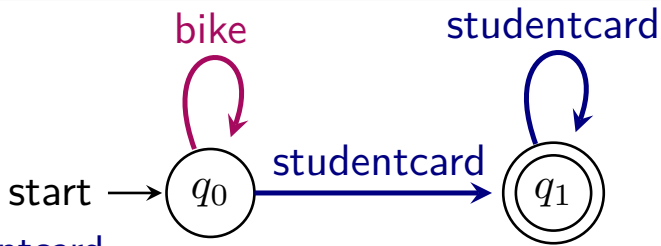


Regular Path Query

Represented by a regular expression R or an automaton A.

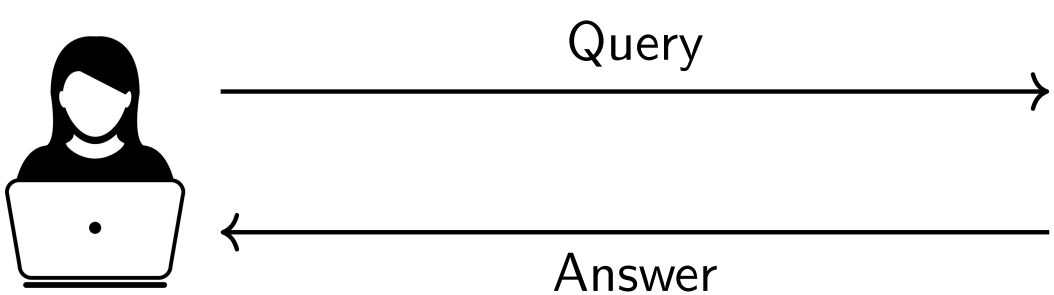
Query: From home to university

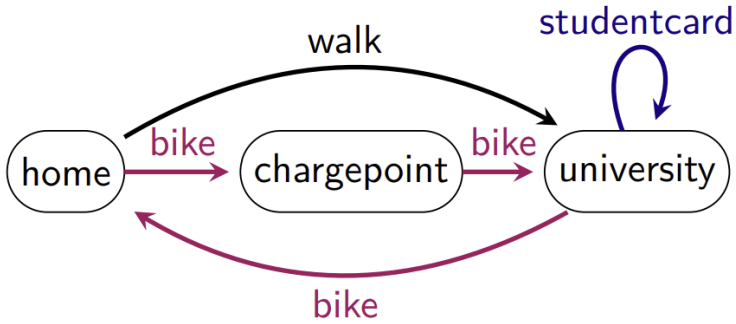
$$R = \text{bike}^* \cdot \text{studentcard}^+$$



Answer: home $\xrightarrow{\text{bike}}$ chargepoint $\xrightarrow{\text{bike}}$ university $\xrightarrow{\text{studentcard}}$ university

Regular Path Queries



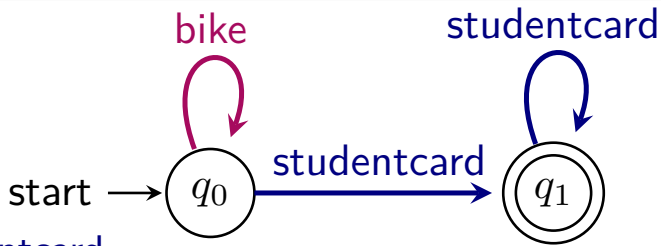


Regular Path Query

Represented by a regular expression R or an automaton A.

Query: From home to university

$$R = \text{bike}^* \cdot \text{studentcard}^+$$

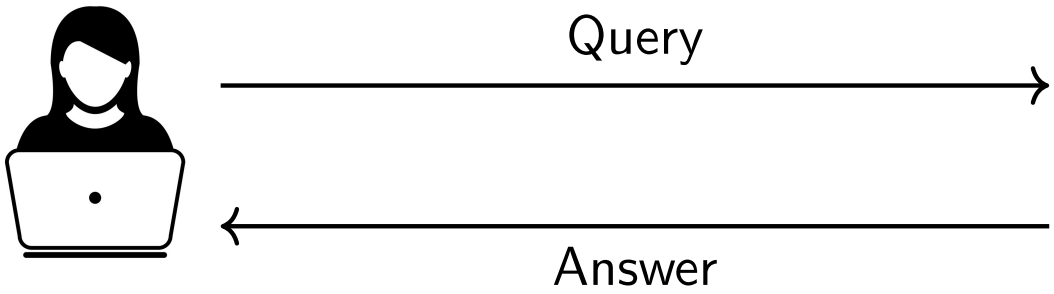


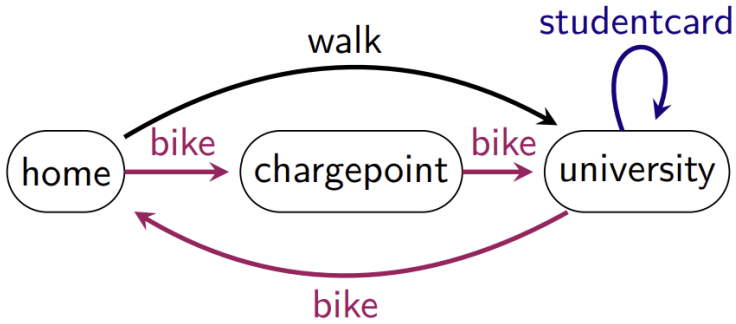
Answer: home $\xrightarrow{\text{bike}}$ chargepoint $\xrightarrow{\text{bike}}$ university $\xrightarrow{\text{studentcard}}$ university

Walk

Alternating sequence of vertices and edges.

Regular Path Queries



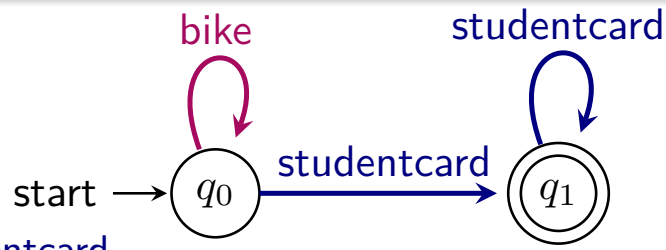


Regular Path Query

Represented by a regular expression R or an automaton A.

Query: From home to university

$$R = \text{bike}^* \cdot \text{studentcard}^+$$



Answer: home $\xrightarrow{\text{bike}}$ chargepoint $\xrightarrow{\text{bike}}$ university $\xrightarrow{\text{studentcard}}$ university

$$\textsc{Match}_{\mathcal{A}}(D)$$

Walk

Alternating sequence of vertices and edges.

Product Graph

$D \times A$

Product of the graph database D and the automaton A representing the RPQ.

Product Graph

$D \times A$

Product of the graph database D and the automaton A representing the RPQ.

$R = \text{bike}^* \cdot \text{studentcard}^+$

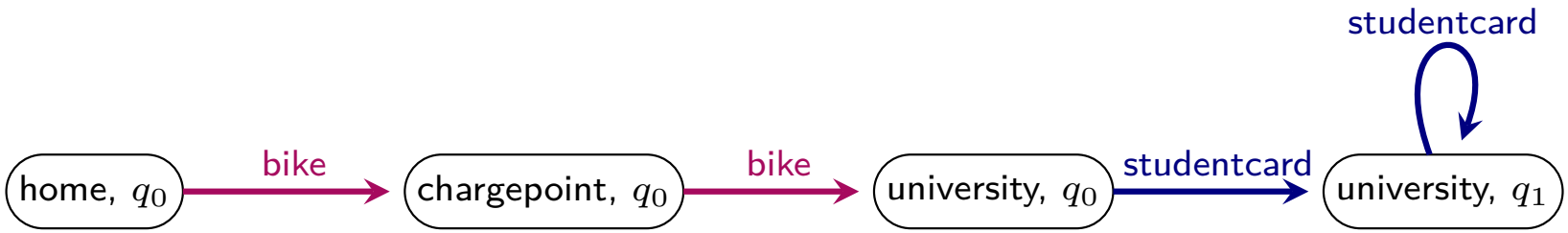


Figure: Part of the Product Graph

Product Graph

$D \times A$

Product of the graph database D and the automaton A representing the RPQ.

$R = \text{bike}^* \cdot \text{studentcard}^+$

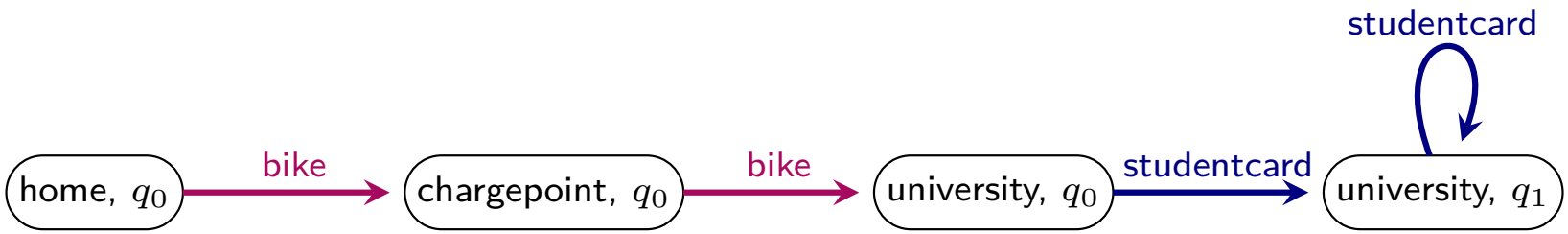


Figure: Part of the Product Graph

- **Run:** walk in the product graph $D \times A$.
  (home, q_0) $\xrightarrow{\text{bike}}$ (chargepoint, q_0) $\xrightarrow{\text{bike}}$ (university, q_0) $\xrightarrow{\text{studentcard}}$ (university, q_1)

Product Graph

$D \times A$

Product of the graph database D and the automaton A representing the RPQ.

$R = \text{bike}^* \cdot \text{studentcard}^+$

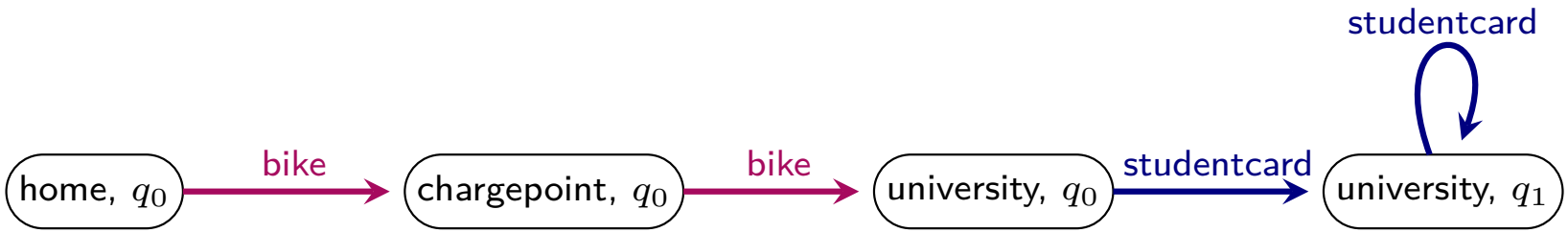


Figure: Part of the Product Graph

- **Run:** walk in the product graph $D \times A$.
  (home, q_0) $\xrightarrow{\text{bike}}$ (chargepoint, q_0) $\xrightarrow{\text{bike}}$ (university, q_0) $\xrightarrow{\text{studentcard}}$ (university, q_1)

- **Projection:** A walk $\in \text{MATCH}_{\mathcal{A}}(D)$.
  home $\xrightarrow{\text{bike}}$ chargepoint $\xrightarrow{\text{bike}}$ university $\xrightarrow{\text{studentcard}}$ university

Matches

Query: From home to university

$R = \text{bike}^* \cdot \text{studentcard}^+$

Answer:

Matches

Query: From home to university

$R = \text{bike}^* \cdot \text{studentcard}^+$

Answer:

- home $\xrightarrow{\text{bike}}$ chargepoint $\xrightarrow{\text{bike}}$ university $\xrightarrow{\text{studentcard}}$ university
- home $\xrightarrow{\text{bike}}$ chargepoint $\xrightarrow{\text{bike}}$ university $\xrightarrow{\text{studentcard}}$ university $\xrightarrow{\text{studentcard}}$ university
- home $\xrightarrow{\text{bike}}$ chargepoint $\xrightarrow{\text{bike}}$ university $\xrightarrow{\text{studentcard}}$ university $\xrightarrow{\text{studentcard}}$ university $\xrightarrow{\text{studentcard}}$ university
- home $\xrightarrow{\text{bike}}$ chargepoint $\xrightarrow{\text{bike}}$ university $\xrightarrow{\text{studentcard}}$ university $\xrightarrow{\text{studentcard}}$ university $\xrightarrow{\text{studentcard}}$ university $\xrightarrow{\text{studentcard}}$ university
- ...

Matches

Query: From home to university

$R = \text{bike}^* \cdot \text{studentcard}^+$

Answer:

- home $\xrightarrow{\text{bike}}$ chargepoint $\xrightarrow{\text{bike}}$ university $\xrightarrow{\text{studentcard}}$ university
- home $\xrightarrow{\text{bike}}$ chargepoint $\xrightarrow{\text{bike}}$ university $\xrightarrow{\text{studentcard}}$ university $\xrightarrow{\text{studentcard}}$ university
- home $\xrightarrow{\text{bike}}$ chargepoint $\xrightarrow{\text{bike}}$ university $\xrightarrow{\text{studentcard}}$ university $\xrightarrow{\text{studentcard}}$ university $\xrightarrow{\text{studentcard}}$ university
- home $\xrightarrow{\text{bike}}$ chargepoint $\xrightarrow{\text{bike}}$ university $\xrightarrow{\text{studentcard}}$ university $\xrightarrow{\text{studentcard}}$ university $\xrightarrow{\text{studentcard}}$ university $\xrightarrow{\text{studentcard}}$ university
- ...

The bag of matches.

Matches

Query: From home to university

$R = \text{bike}^* \cdot \text{studentcard}^+$

Answer:

- home $\xrightarrow{\text{bike}}$ chargepoint $\xrightarrow{\text{bike}}$ university $\xrightarrow{\text{studentcard}}$ university
- home $\xrightarrow{\text{bike}}$ chargepoint $\xrightarrow{\text{bike}}$ university $\xrightarrow{\text{studentcard}}$ university $\xrightarrow{\text{studentcard}}$ university
- home $\xrightarrow{\text{bike}}$ chargepoint $\xrightarrow{\text{bike}}$ university $\xrightarrow{\text{studentcard}}$ university $\xrightarrow{\text{studentcard}}$ university $\xrightarrow{\text{studentcard}}$ university
- home $\xrightarrow{\text{bike}}$ chargepoint $\xrightarrow{\text{bike}}$ university $\xrightarrow{\text{studentcard}}$ university $\xrightarrow{\text{studentcard}}$ university $\xrightarrow{\text{studentcard}}$ university $\xrightarrow{\text{studentcard}}$ university
- ...

The bag of matches.

Issue? Infinite matches!

RPQ Semantics

Compute a finite and meaningful output from the matches.

Output of a semantics X: $[\![\mathcal{A}]\!]_X(D)$

RPQ Semantics

Compute a finite and meaningful output from the matches.

Output of a semantics X: $[\![\mathcal{A}]\!]_X(D)$

Criteria

- Coverage: Not defined in literature.
- Computational Properties: Complexity of the most common computational problems in database querying.

Motivations

Motivations

- Trail Semantics: Used in Cypher (Neo4j).

Motivations

- Trail Semantics: Used in Cypher (Neo4j).
  -> May discard valuable matches.
  -> Computational problems' complexity.

Motivations

- Trail Semantics: Used in Cypher (Neo4j).
  -> May discard valuable matches.
  -> Computational problems' complexity.

- Shortest-walk Semantics: Used in GSQL (TigerGraph) and PGQL (Oracle).

Motivations

- Trail Semantics: Used in Cypher (Neo4j).
  -> May discard valuable matches.
  -> Computational problems' complexity.

- Shortest-walk Semantics: Used in GSQL (TigerGraph) and PGQL (Oracle).
  -> Length is not always meaningful.
  -> Not representative of data.

Motivations

- Trail Semantics: Used in Cypher (Neo4j).
  -> May discard valuable matches.
  -> Computational problems' complexity.

- Shortest-walk Semantics: Used in GSQL (TigerGraph) and PGQL (Oracle).
  -> Length is not always meaningful.
  -> Not representative of data.

We need new candidates.

New RPQ Semantics

Shortest-coverage Semantics

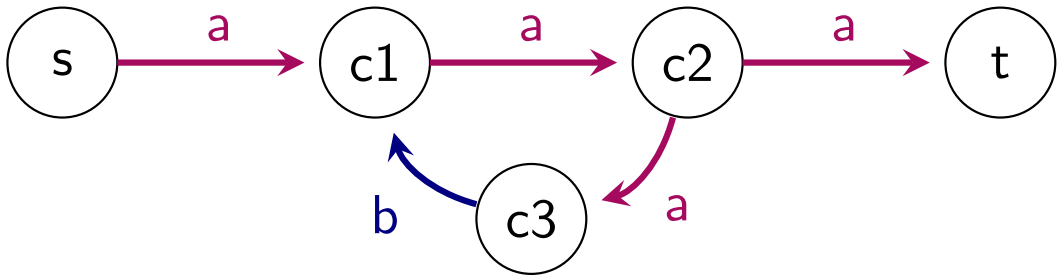
Output

$$[\![\mathcal{A}]\!]_{SC}(D) = \bigcup_{(s,t,v)\in V^3} \{\!\{w \in \textsc{Match}_{\mathcal{A}}(D,s,t) \mid w \text{ is a shortest walk covering } v \text{ in } \textsc{Match}_{\mathcal{A}}(D,s,t)\}\!\}$$

Shortest-coverage Semantics

Output

$$[\![\mathcal{A}]\!]_{SC}(D) = \bigcup_{(s,t,v)\in V^3} \{\!\{w \in \textsc{Match}_{\mathcal{A}}(D,s,t) \mid w \text{ is a shortest walk covering } v \text{ in } \textsc{Match}_{\mathcal{A}}(D,s,t)\}\!\}$$

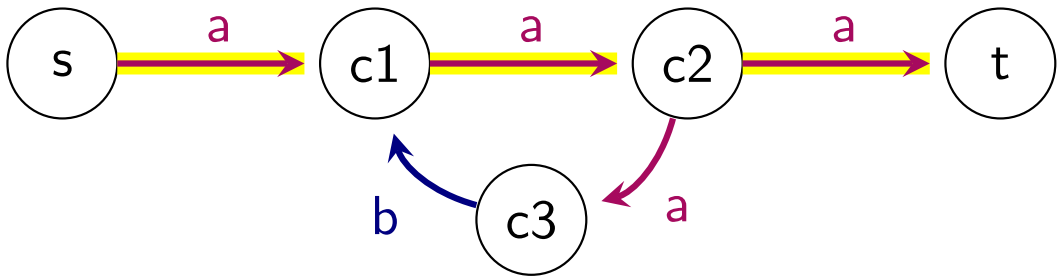


Query: From s to t $\quad R = (a+b)^*$

Shortest-coverage Semantics

Output

$$[\![\mathcal{A}]\!]_{SC}(D) = \bigcup_{(s,t,v)\in V^3} \{\!\{w \in \textsc{Match}_{\mathcal{A}}(D,s,t) \mid w \text{ is a shortest walk covering } v \text{ in } \textsc{Match}_{\mathcal{A}}(D,s,t)\}\!\}$$



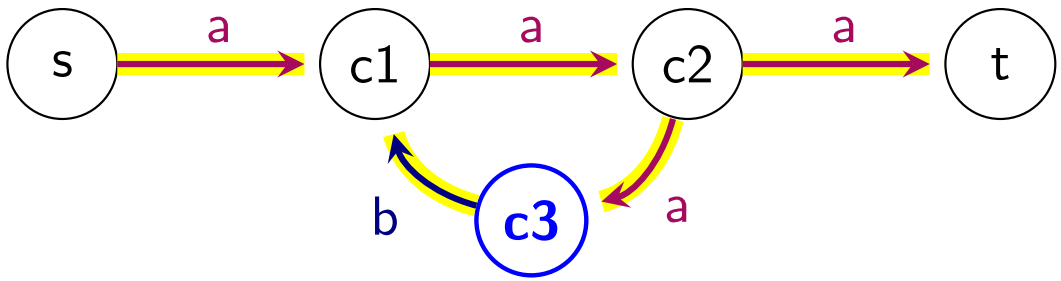
Query: From s to t $\quad R = (a+b)^*$

- **For s, c1, c2, t:** s $\xrightarrow{a}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ t

Shortest-coverage Semantics

Output

$$[\![\mathcal{A}]\!]_{SC}(D) = \bigcup_{(s,t,v)\in V^3} \{\!\{w \in \textsc{Match}_{\mathcal{A}}(D,s,t) \mid w \text{ is a shortest walk covering } v \text{ in } \textsc{Match}_{\mathcal{A}}(D,s,t)\}\!\}$$



Query: From s to t $\quad R = (a+b)^*$

- For s, c1, c2, t: s $\xrightarrow{a}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ t
- **For c3:** s $\xrightarrow{a}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ c3 $\xrightarrow{b}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ t

Minimal-walk Semantics

In a bag of walks S where $w, w' \in S$,

Minimal-walk Semantics

In a bag of walks S where $w, w' \in S$,

Subwalk Order

$w' \sqsubseteq w \iff w' \ll^* w.$

Minimal-walk Semantics

In a bag of walks S where $w, w' \in S$,

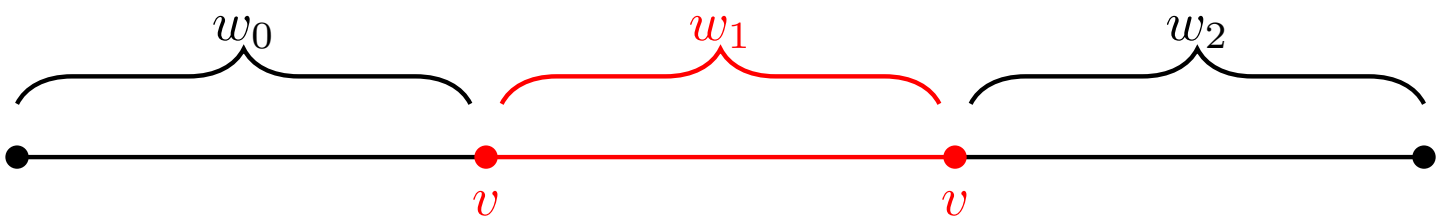
Subwalk Order

$w' \sqsubseteq w \iff w' \ll^* w.$

Removal Operation

$w'' \ll w \iff \exists w_0, w_1, w_2, w = w_0 \cdot w_1 \cdot w_2$ and $w'' = w_0 \cdot w_2$.

w:



Minimal-walk Semantics

In a bag of walks S where $w, w' \in S$,

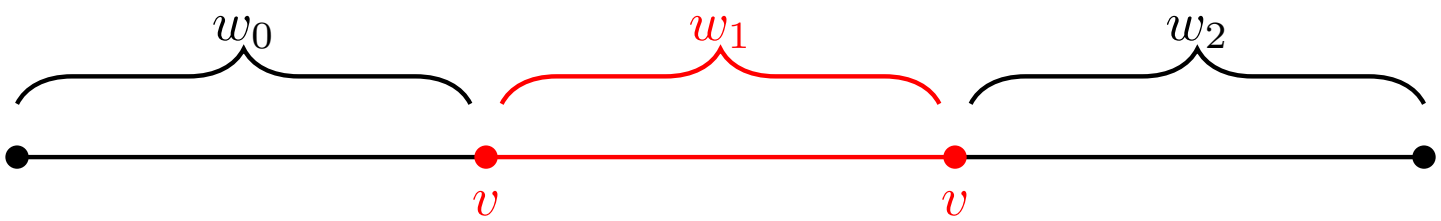
Subwalk Order

$w' \sqsubseteq w \iff w' \ll^* w.$

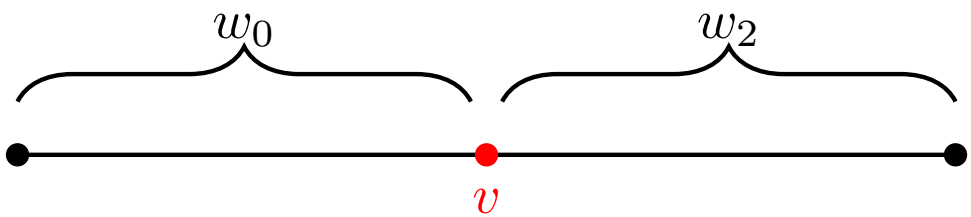
Removal Operation

$w'' \ll w \iff \exists w_0, w_1, w_2, w = w_0 \cdot w_1 \cdot w_2$ and $w'' = w_0 \cdot w_2$.

w:



w'':



Minimal-walk Semantics

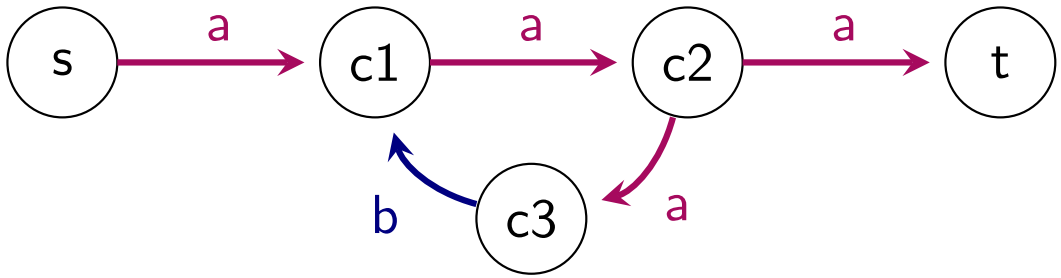
Output

$$[\![\mathcal{A}]\!]_{MW}(D) = \{\!\{w \in \textsc{Match}_{\mathcal{A}}(D) \mid w \text{ is minimal in } \textsc{Match}_{\mathcal{A}}(D)\}\!\}$$

Minimal-walk Semantics

Output

$$[\![\mathcal{A}]\!]_{MW}(D) = \{\!\{w \in \textsc{Match}_{\mathcal{A}}(D) \mid w \text{ is minimal in } \textsc{Match}_{\mathcal{A}}(D)\}\!\}$$

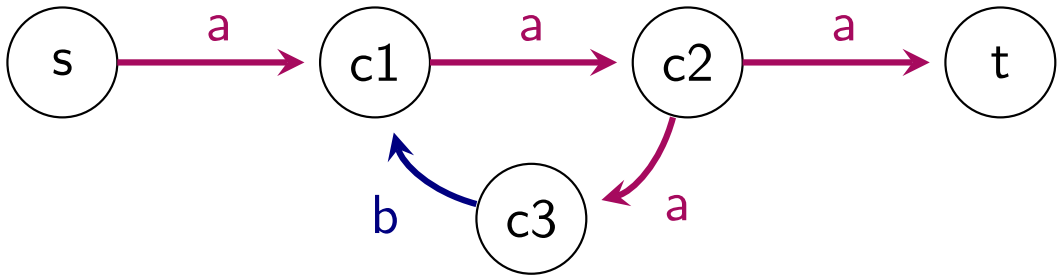


Query: From s to t $\qquad R = (a + b)^*$

Minimal-walk Semantics

Output

$$[\![\mathcal{A}]\!]_{MW}(D) = \{\!\{w \in \text{MATCH}_{\mathcal{A}}(D) \mid w \text{ is minimal in } \text{MATCH}_{\mathcal{A}}(D)\}\!\}$$



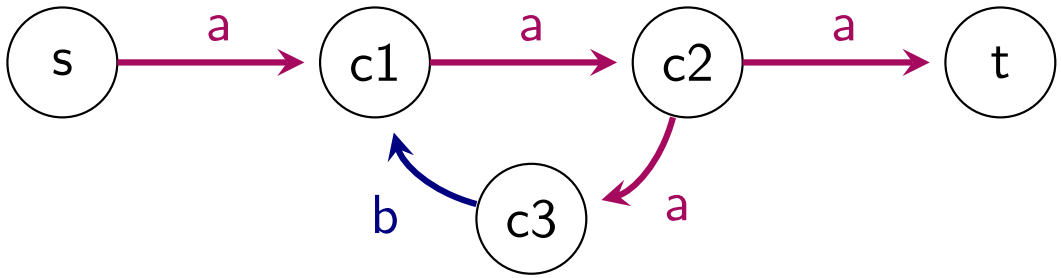
Query: From s to t $\quad R = (a+b)^*$

w: s $\xrightarrow{a}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ c3 $\xrightarrow{b}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ t

Minimal-walk Semantics

Output

$$[\![\mathcal{A}]\!]_{MW}(D) = \{\!\{w \in \textsc{Match}_{\mathcal{A}}(D) \mid w \text{ is minimal in } \textsc{Match}_{\mathcal{A}}(D)\}\!\}$$



Query: From s to t $\quad R = (a+b)^*$

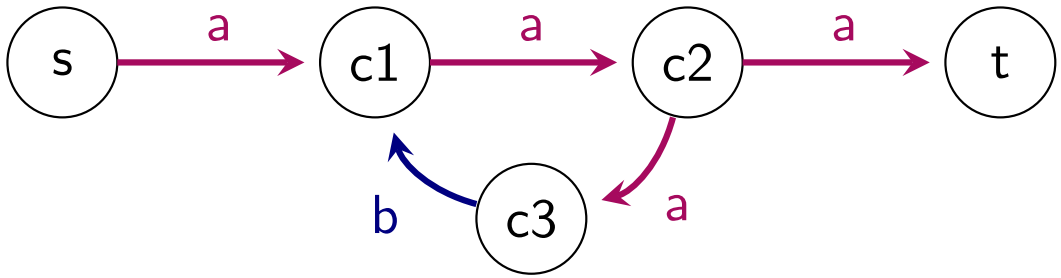
w: s $\xrightarrow{a}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ c3 $\xrightarrow{b}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ t

Not a minimal walk!

Minimal-walk Semantics

Output

$$[\![\mathcal{A}]\!]_{MW}(D) = \{\!\{w \in \text{MATCH}_{\mathcal{A}}(D) \mid w \text{ is minimal in } \text{MATCH}_{\mathcal{A}}(D)\}\!\}$$



Query: From s to t $\quad R = (a+b)^*$

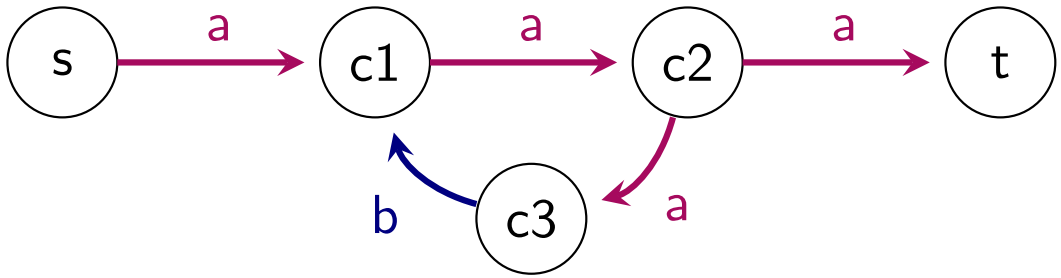
w: s $\xrightarrow{a}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ c3 $\xrightarrow{b}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ t

Not a minimal walk! It has a subwalk that is a match.

Minimal-walk Semantics

Output

$$[\![\mathcal{A}]\!]_{MW}(D) = \{\!\{w \in \textsc{Match}_{\mathcal{A}}(D) \mid w \text{ is minimal in } \textsc{Match}_{\mathcal{A}}(D)\}\!\}$$



Query: From s to t $\quad R = (a+b)^*$

w: s $\xrightarrow{a}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ c3 $\xrightarrow{b}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ t

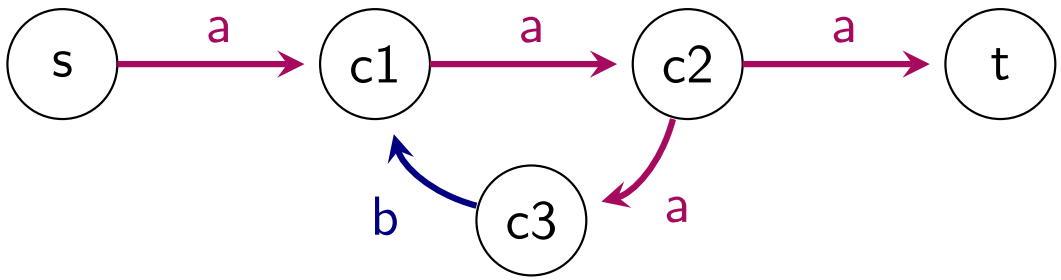
Not a minimal walk! It has a subwalk that is a match.

w': s $\xrightarrow{a}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ t

Minimal-walk Semantics

Output

$$[\![\mathcal{A}]\!]_{MW}(D) = \{\!\{w \in \textsc{Match}_{\mathcal{A}}(D) \mid w \text{ is minimal in } \textsc{Match}_{\mathcal{A}}(D)\}\!\}$$



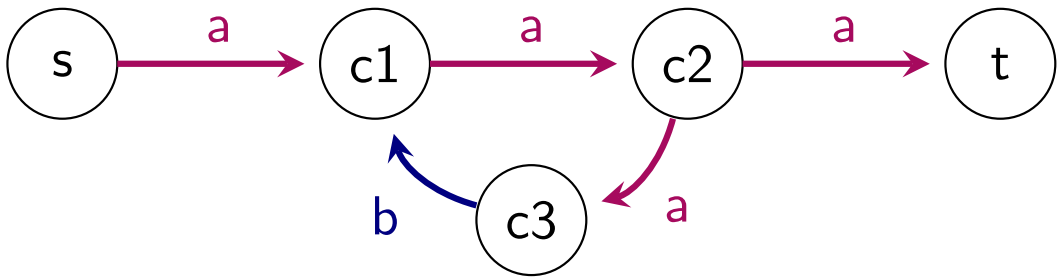
Query: From s to t $\qquad R = a^* b\ a^*$

w: s $\xrightarrow{a}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ c3 $\xrightarrow{b}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ t

Minimal-walk Semantics

Output

$$[\![\mathcal{A}]\!]_{MW}(D) = \{\!\{w \in \text{MATCH}_{\mathcal{A}}(D) \mid w \text{ is minimal in } \text{MATCH}_{\mathcal{A}}(D)\}\!\}$$



Query: From s to t $\quad R = a^* b\ a^*$

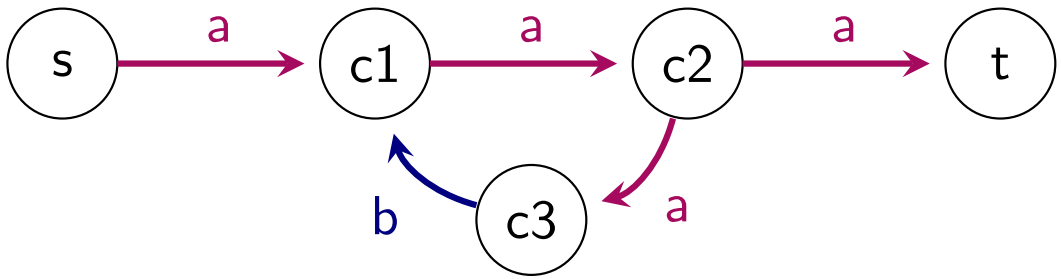
w: s $\xrightarrow{a}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ c3 $\xrightarrow{b}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ t

It is a minimal walk.

Minimal-walk Semantics

Output

$$[\![\mathcal{A}]\!]_{MW}(D) = \{\!\{w \in \text{MATCH}_{\mathcal{A}}(D) \mid w \text{ is minimal in } \text{MATCH}_{\mathcal{A}}(D)\}\!\}$$



Query: From s to t $\quad R = a^* b\ a^*$

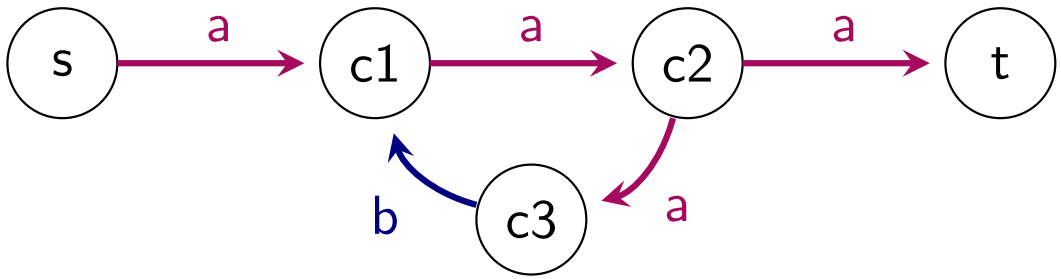
w: s $\xrightarrow{a}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ c3 $\xrightarrow{b}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ t

It is a minimal walk. No subwalks in the matches!

Minimal-walk Semantics

Output

$$[\![\mathcal{A}]\!]_{MW}(D) = \{\!\{w \in \text{MATCH}_{\mathcal{A}}(D) \mid w \text{ is minimal in } \text{MATCH}_{\mathcal{A}}(D)\}\!\}$$



Query: From s to t $\quad R = a^* b\ a^*$

w: s $\xrightarrow{a}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ c3 $\xrightarrow{b}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ t

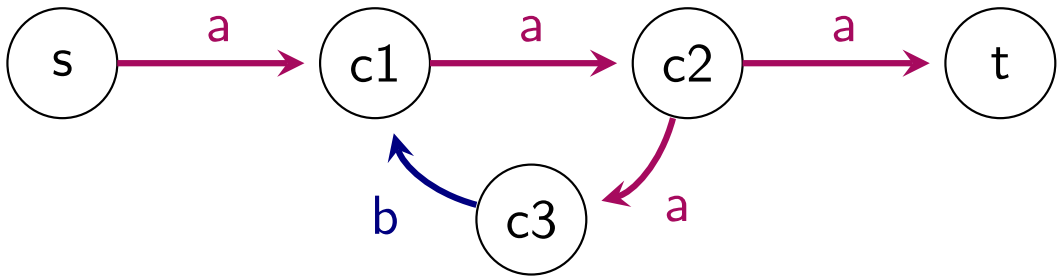
It is a minimal walk. No subwalks in the matches!

w': s $\xrightarrow{a}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ t

Minimal-walk Semantics

Output

$$[\![\mathcal{A}]\!]_{MW}(D) = \{\!\{w \in \textsc{Match}_{\mathcal{A}}(D) \mid w \text{ is minimal in } \textsc{Match}_{\mathcal{A}}(D)\}\!\}$$



Query: From s to t $\quad R = a^* b\ a^*$

w: s $\xrightarrow{a}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ c3 $\xrightarrow{b}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ t

It is a minimal walk. No subwalks in the matches!

w': s $\xrightarrow{a}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ t Not a match!

Computational Problems: Walk Membership

Problem definition

Walk Membership Under X Semantics

Input: A graph database D, a query $\mathcal{A}$, a walk $w \in \text{Match}_{\mathcal{A}}(D)$.
Output: $w \in [\![\mathcal{A}]\!]_X(D)$?

Walk Membership under shortest-coverage semantics

Theorem

Walk Membership *under shortest-coverage semantics is in* ***P***.

Walk Membership under shortest-coverage semantics

Theorem

Walk Membership *under shortest-coverage semantics is in* ***P***.

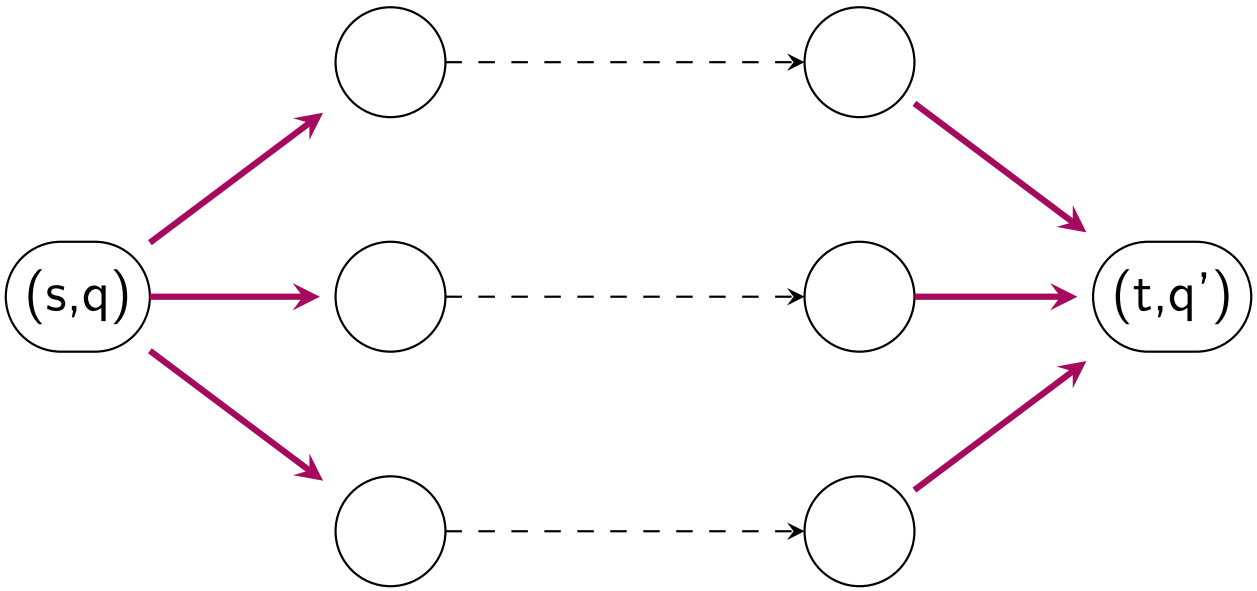
Proof. We work on the product graph.

Walk Membership under shortest-coverage semantics

Theorem

Walk Membership *under shortest-coverage semantics is in* ***P****.*

Proof.



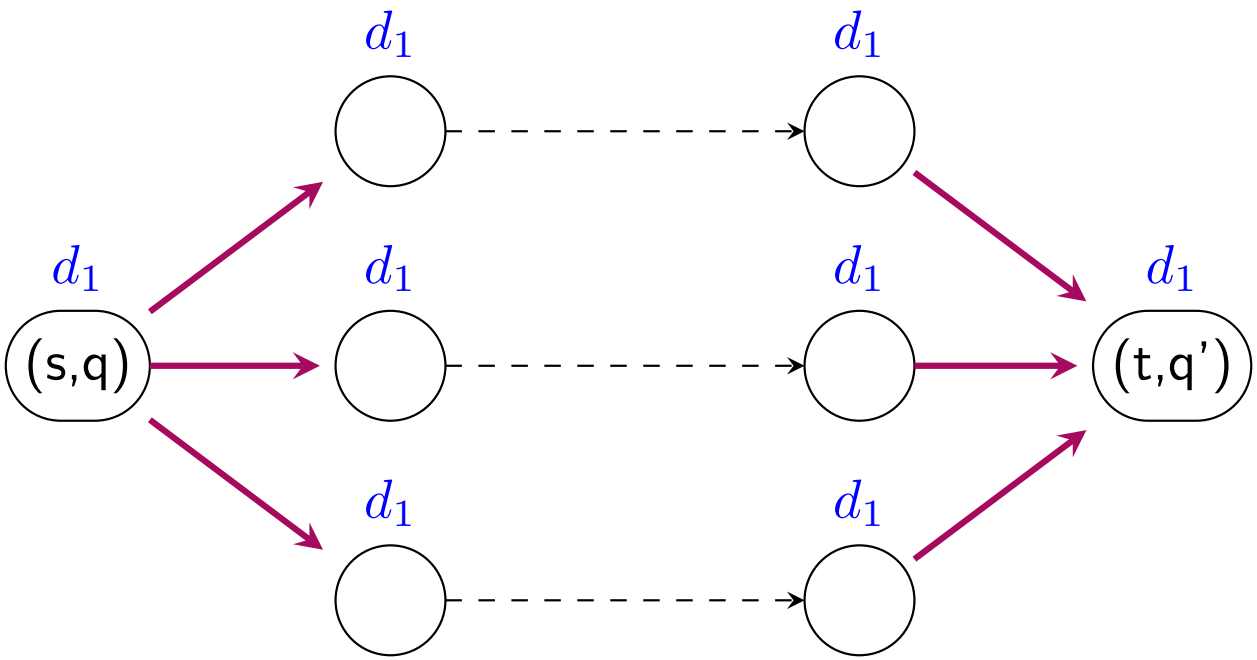
1. BFS on $D \times \mathcal{A}$ from sources to targets.

Walk Membership under shortest-coverage semantics

Theorem

Walk Membership *under shortest-coverage semantics is in* ***P****.*

Proof.



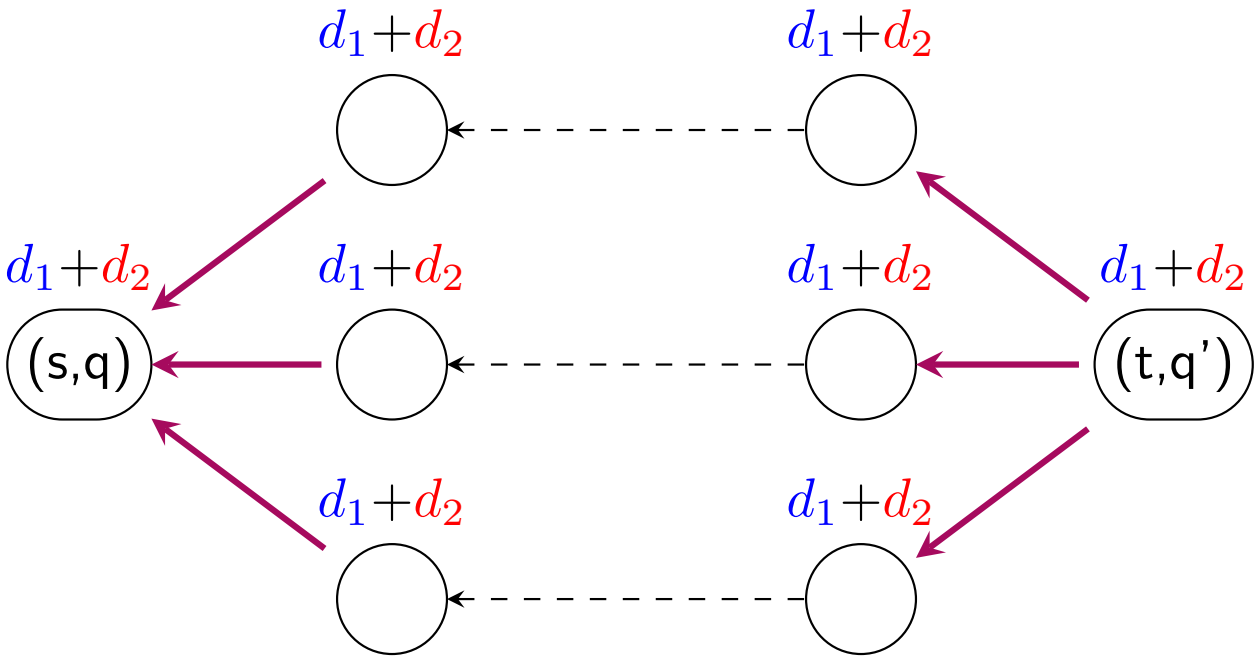
1. BFS on $D \times \mathcal{A}$ from sources to each vertex. Distances d_1

Walk Membership under shortest-coverage semantics

Theorem

Walk Membership *under shortest-coverage semantics is in* ***P****.*

Proof.



1. BFS on $D \times \mathcal{A}$ from sources to each vertex. Distances d_1
2. BFS on $(D \times \mathcal{A})^T$ from targets to each vertex. Mirror Distances d_2

Walk Membership under shortest-coverage semantics

Theorem

Walk Membership *under shortest-coverage semantics is in* ***P***.

Proof.

3. $\exists v$ a vertex in w such that $\forall q \in Q, (d_1 + d_2)((v, q)) \geq |w|$? $w \in [\![\mathcal{A}]\!]_{SC}(D)$

Walk Membership under shortest-coverage semantics

Theorem

WALK MEMBERSHIP *under shortest-coverage semantics is in* ***P***.

Proof.

3. $\exists v$ a vertex in w such that $\forall q \in Q, (d_1 + d_2)((v, q)) \geq |w|$? $w \in [\![\mathcal{A}]\!]_{SC}(D)$

   If no such vertex exists: $w \notin [\![\mathcal{A}]\!]_{SC}(D)$

Walk Membership under minimal-walk semantics

Theorem

Walk Membership *under minimal-walk semantics is in* ***P****.*

Walk Membership under minimal-walk semantics

Theorem

Walk Membership *under minimal-walk semantics is in* ***P****.*

Proof. We construct a new structure: the *subwalk automaton* $D_{\sqsubseteq w}$.

Walk Membership under minimal-walk semantics

Theorem

Walk Membership *under minimal-walk semantics is in* ***P****.*

Proof. We construct a new structure: the *subwalk automaton* $D_{\sqsubseteq w}$.

w: s $\xrightarrow{a}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ c1 $\xrightarrow{a}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ t

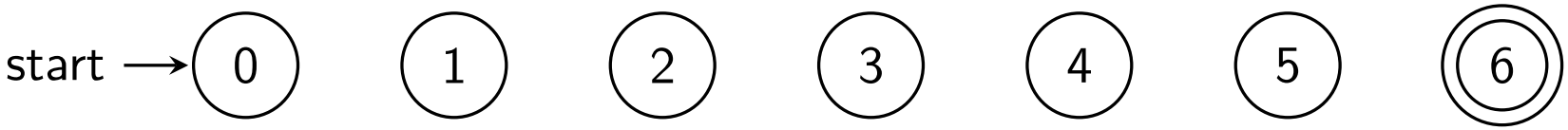
Walk Membership under minimal-walk semantics

Theorem

Walk Membership *under minimal-walk semantics is in* ***P****.*

Proof. We construct a new structure: the *subwalk automaton* $D_{\sqsubseteq w}$.

w: s $\xrightarrow{a}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ c1 $\xrightarrow{a}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ t



- $Q = \{i \mid 0 \leq i \leq |w| + 1\}$, $I = \{0\}$, $F = \{|w| + 1\}$.

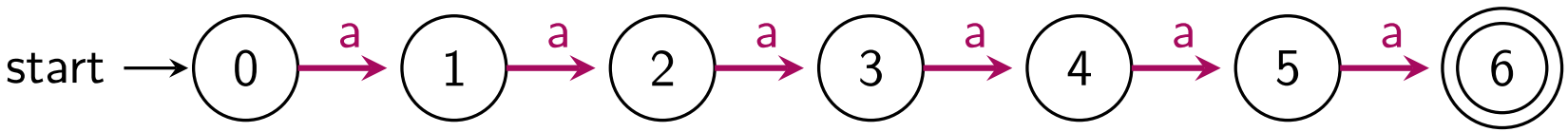
Walk Membership under minimal-walk semantics

Theorem

WALK MEMBERSHIP *under minimal-walk semantics is in* ***P****.*

Proof. We construct a new structure: the *subwalk automaton* $D_{\sqsubseteq w}$.

w: s $\xrightarrow{a}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ c1 $\xrightarrow{a}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ t



- $\delta = \{(i, a, i+1) \mid 0 \leq i \leq |w|, a \in \mathcal{A}\}$.

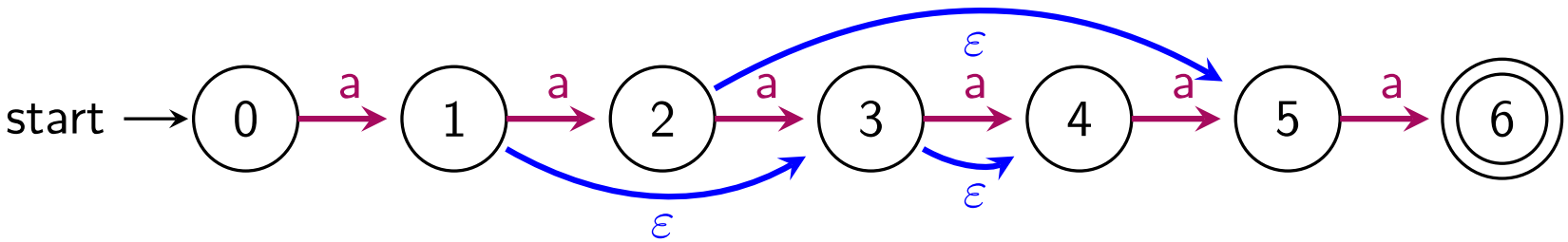
Walk Membership under minimal-walk semantics

Theorem

WALK MEMBERSHIP *under minimal-walk semantics is in* ***P****.*

Proof. We construct a new structure: the *subwalk automaton* $D_{\sqsubseteq w}$.

w: s $\xrightarrow{a}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ c1 $\xrightarrow{a}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ t



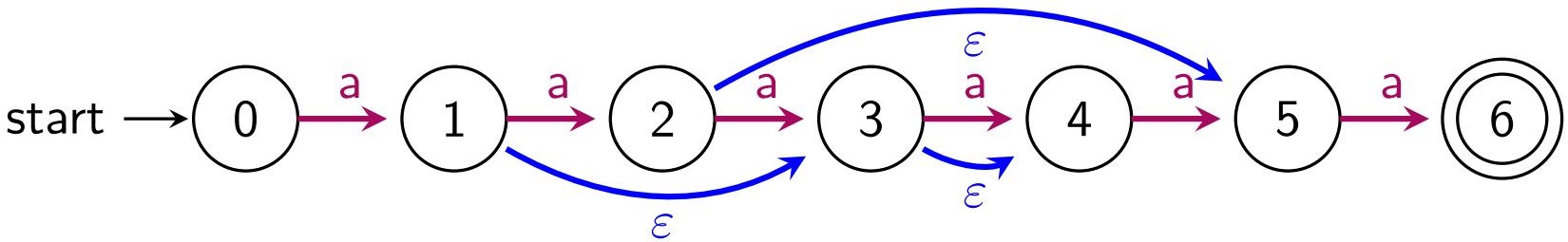
- $\delta = \{(i, a, i+1) \mid 0 \leq i \leq |w|+1, a \in \mathcal{A}\}$
  $\cup \quad \{(i, \varepsilon, j) \mid n_i = n_j, 0 \leq i < j \leq |w|+1, \nexists l \in]i, j[, n_l = n_i\}$.

Walk Membership under minimal-walk semantics

Theorem

WALK MEMBERSHIP *under minimal-walk semantics is in* ***P****.*

Proof. We construct a new structure: the *subwalk automaton* $D_{\sqsubseteq w}$.
w: s $\xrightarrow{a}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ c1 $\xrightarrow{a}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ t



Lemma

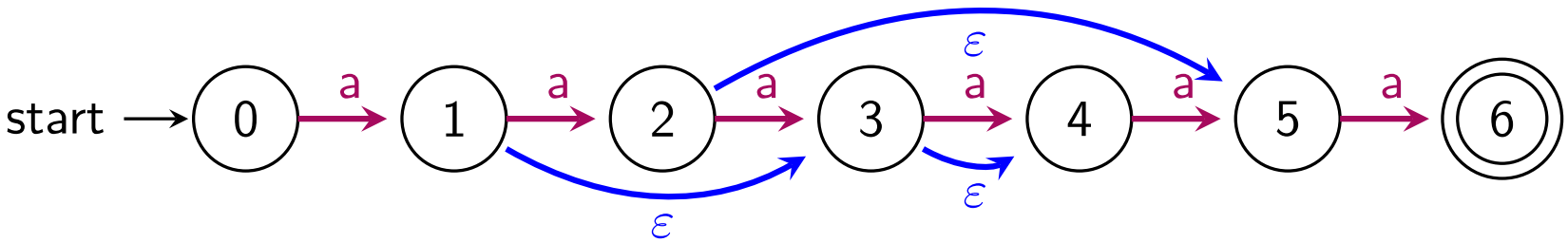
$D_{\sqsubseteq w}$ has a size in $O(|w|)$ and can be constructed in $O(|w| \log |w|)$.

Walk Membership under minimal-walk semantics

Theorem

WALK MEMBERSHIP *under minimal-walk semantics is in* ***P****.*

Proof. We construct a new structure: the *subwalk automaton* $D_{\sqsubseteq w}$.
w: s $\xrightarrow{a}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ c1 $\xrightarrow{a}$ c1 $\xrightarrow{a}$ c2 $\xrightarrow{a}$ t



Lemma

$D_{\sqsubseteq w}$ *has a size in* $O(|w|)$ *and can be constructed in* $O(|w| \log |w|)$.

Lemma

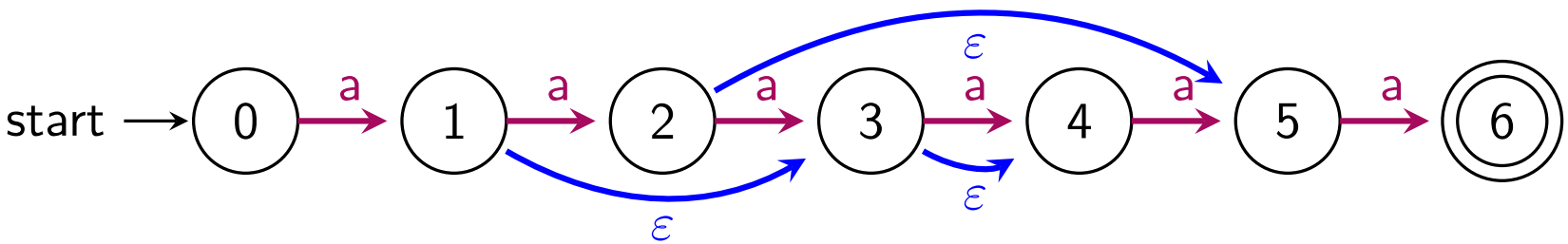
$\mathcal{L}(D_{\sqsubseteq w}) = \{\text{LBL}(w') \mid w' \sqsubset w\}$.

Walk Membership under minimal-walk semantics

Theorem

WALK MEMBERSHIP *under minimal-walk semantics is in* ***P****.*

Proof. We construct a new structure: the *subwalk automaton* $D_{\sqsubseteq w}$.



- Using a BFS on $D_{\sqsubseteq w} \times \mathcal{A}$, we check if $|w| = |shortestwalk|$?
  $w \in [\![\mathcal{A}]\!]_{MW}(D)$
  Otherwise, w is not a minimal walk. $w \notin [\![\mathcal{A}]\!]_{MW}(D)$

Computational Problems: Walk Enumeration

Problem definition

Walk Enumeration Under X Semantics

Input: A graph database D, a query $\mathcal{A}$, (s, t).
Output: $\{\!\{w \in [\![\mathcal{A}]\!]_X(D) \mid \textsc{EndPoints}(w) = (s, t)\}\!\}$.

Enumeration Complexity

Enumeration Complexity

- The output is usually exponentially larger than the input.

Enumeration Complexity

- The output is usually exponentially larger than the input.

Consequence

The classical complexity is not informative enough. Other parameters are needed!

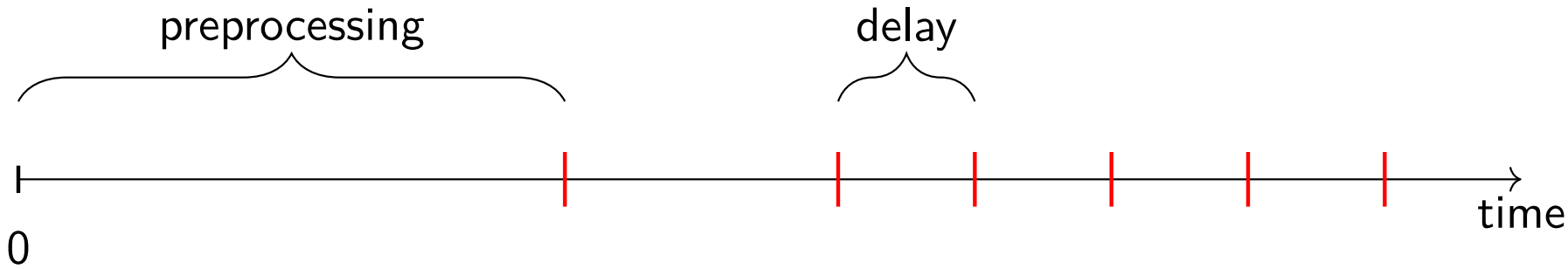
Enumeration Complexity

- The output is usually exponentially larger than the input.

Consequence

The classical complexity is not informative enough. Other parameters are needed!

1. The preprocessing: the time before outputting the first solution.
2. The delay: the time between two consecutive solutions.



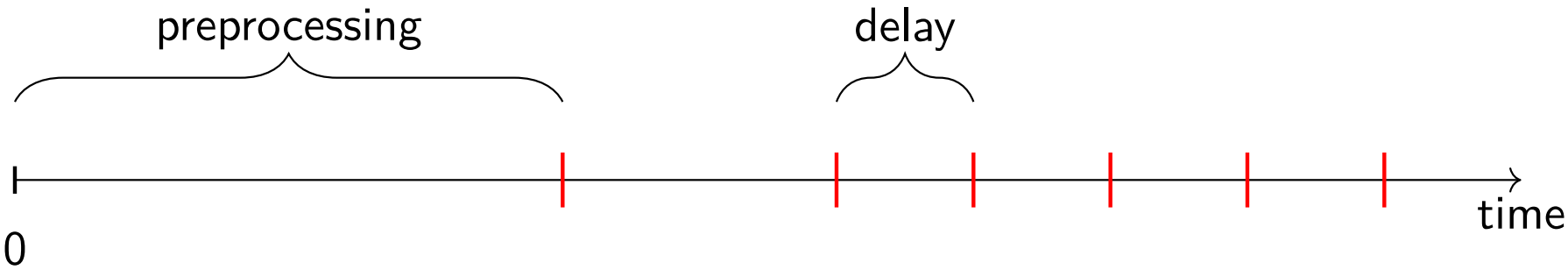
Enumeration Complexity

- The output is usually exponentially larger than the input.

Consequence

The classical complexity is not informative enough. Other parameters are needed!

1. The preprocessing: the time before outputting the first solution.
2. The delay: the time between two consecutive solutions.



Definition

A problem A is in *DelayP* if there is a polynomial p and an algorithm which solves A on input x with a polynomial preprocessing and a delay in $O(p(|x|))$.

Walk Enumeration under shortest-coverage semantics

Theorem

Walk Enumeration *under shortest-coverage semantics is in* ***DelayP***.

Walk Enumeration under shortest-coverage semantics

Theorem

Walk Enumeration *under shortest-coverage semantics is in* ***DelayP***.

Proof. We work on the product graph.

Walk Enumeration under shortest-coverage semantics

Theorem

Walk Enumeration *under shortest-coverage semantics is in* ***DelayP***.

Proof. We work on the product graph.

1. Use Floyd-Warshall to get the distances between all pairs of vertices.

Walk Enumeration under shortest-coverage semantics

Theorem

Walk Enumeration *under shortest-coverage semantics is in* ***DelayP***.

Proof. We work on the product graph.

1. Use Floyd-Warshall to get the distances between all pairs of vertices.
2. For each vertex, find the states i, q, f that minimize the distance: $d((s,i),(v,q)) + d((v,q),(t,f))$.

Walk Enumeration under shortest-coverage semantics

Theorem

Walk Enumeration *under shortest-coverage semantics is in* ***DelayP***.

Proof. We work on the product graph.

1. Use Floyd-Warshall to get the distances between all pairs of vertices.
2. For each vertex, find the states i, q, f that minimize the distance: $d((s,i),(v,q)) + d((v,q),(t,f))$.
   - Get the runs.

Walk Enumeration under shortest-coverage semantics

Theorem

Walk Enumeration *under shortest-coverage semantics is in* ***DelayP***.

Proof. We work on the product graph.

1. Use Floyd-Warshall to get the distances between all pairs of vertices.
2. For each vertex, find the states i, q, f that minimize the distance: $d((s,i),(v,q)) + d((v,q),(t,f))$.
   - Get the runs.
   - Output the walks.

Another Enumeration Problem

Distinct Walk Enumeration Under X Semantics

Input: A graph database D, a query $\mathcal{A}$, (s, t).
Output: $\{w \in [\![\mathcal{A}]\!]_X(D) \mid \textsc{EndPoints}(w) = (s, t)\}$.

Distinct Walk Enumeration

Distinct Walk Enumeration

A general approach: DFS on the graph database.

Distinct Walk Enumeration

A general approach: DFS on the graph database. Using another problem.

Distinct Walk Enumeration

A general approach: DFS on the graph database. Using another problem.

Prolongability

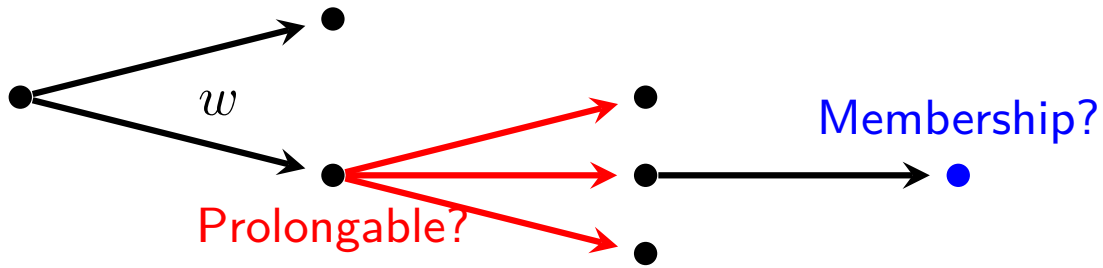
- Given a walk w, decide if w can be prolonged to a solution.

Distinct Walk Enumeration

A general approach: DFS on the graph database. Using another problem.

Prolongability

- Given a walk w, decide if w can be prolonged to a solution.

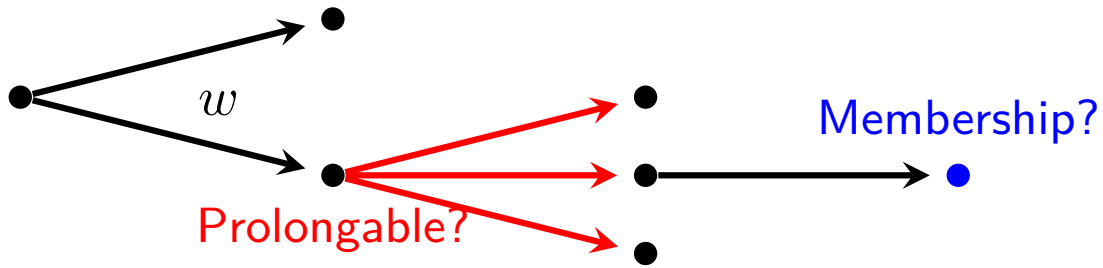


Distinct Walk Enumeration

A general approach: DFS on the graph database. Using another problem.

Prolongability

- Given a walk w, decide if w can be prolonged to a solution.



Theorem

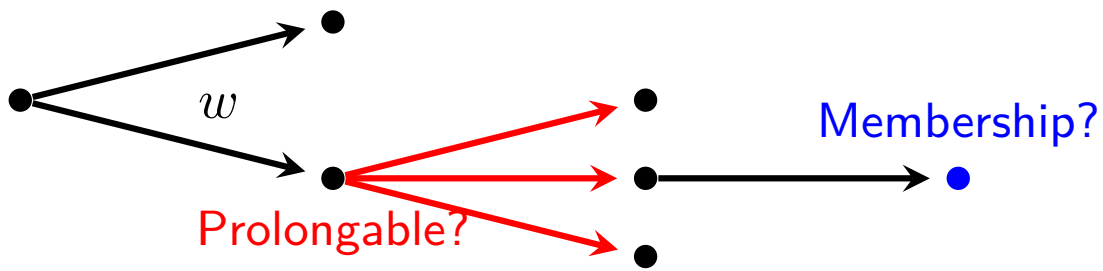
If Prolongability $\in$ ***P*** *and* Walk Membership $\in$ ***P***, *then* Distinct Walk Enumeration $\in$ ***DelayP***.

Distinct Walk Enumeration

A general approach: DFS on the graph database. Using another problem.

Prolongability

- Given a walk w, decide if w can be prolonged to a solution.



Theorem

If Prolongability $\in$ ***P*** *and* Walk Membership $\in$ ***P***, *then* Distinct Walk Enumeration $\in$ ***DelayP***.

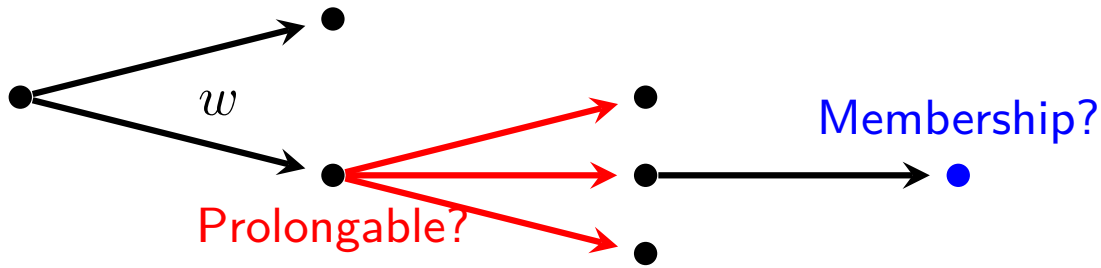
Both problems are in **P** under shortest-coverage semantics.

Distinct Walk Enumeration

A general approach: DFS on the graph database. Using another problem.

Prolongability

- Given a walk w, decide if w can be prolonged to a solution.



Theorem

If Prolongability $\in$ ***P*** *and* Walk Membership $\in$ ***P***, *then* Distinct Walk Enumeration $\in$ ***DelayP***.

Both problems are in **P** under shortest-coverage semantics.

Theorem

Distinct Walk Enumeration *under shortest-coverage semantics* $\in$ ***DelayP***.

(Distinct) Walk Enum. under minimal-walk semantics

(Distinct) Walk Enum. under minimal-walk semantics

Open problem:

Is (Distinct) Walk Enum. under minimal-walk semantics in **DelayP**?

(Distinct) Walk Enum. under minimal-walk semantics

Open problem:

Is (Distinct) Walk Enum. under minimal-walk semantics in **DelayP**?

Theorem

Prolongability *under minimal-walk semantics is NP-complete.*

(Distinct) Walk Enum. under minimal-walk semantics

Open problem:

Is (Distinct) Walk Enum. under minimal-walk semantics in **DelayP**?

Theorem

Prolongability *under minimal-walk semantics is NP-complete.*

Proof. Reduction from Two-Disjoint-Paths.

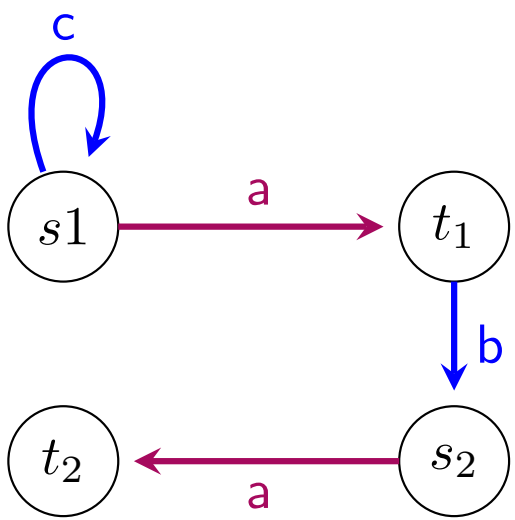
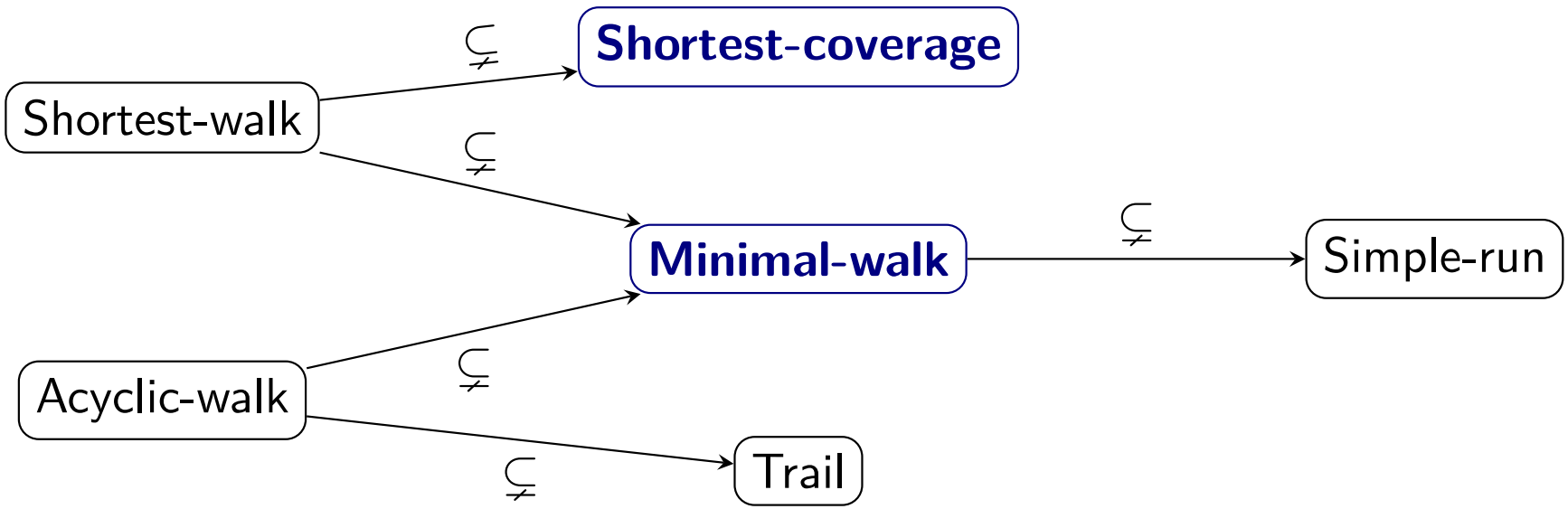


Figure: Reduction with the query $(c \cdot a^* \cdot b \cdot a^*) + a^*$

Conclusion

Summary:

	Shortest-coverage	**Minimal-walk**
Coverage	Vertex coverage	Subwalk coverage
Tuple Membership	NL-complete	NL-complete
Walk Membership	P	P
Walk Enumeration	DelayP	Open
Distinct Walk Enumeration	DelayP	Open



Conclusion

Summary:

	Shortest-coverage	**Minimal-walk**
Coverage	Vertex coverage	Subwalk coverage
Tuple Membership	NL-complete	NL-complete
Walk Membership	P	P
Walk Enumeration	DelayP	Open
Distinct Walk Enumeration	DelayP	Open

Future works:

Conclusion

Summary:

	Shortest-coverage	**Minimal-walk**
Coverage	Vertex coverage	Subwalk coverage
Tuple Membership	NL-complete	NL-complete
Walk Membership	P	P
Walk Enumeration	DelayP	Open
Distinct Walk Enumeration	DelayP	Open

Future works:

- See if some P problems can be P-complete or NL-complete.

Conclusion

Summary:

	Shortest-coverage	**Minimal-walk**
Coverage	Vertex coverage	Subwalk coverage
Tuple Membership	NL-complete	NL-complete
Walk Membership	P	P
Walk Enumeration	DelayP	Open
Distinct Walk Enumeration	DelayP	Open

Future works:

- See if some P problems can be P-complete or NL-complete.
- Static analysis of the semantics: Containment, Equivalence.

Conclusion

Summary:

	Shortest-coverage	**Minimal-walk**
Coverage	Vertex coverage	Subwalk coverage
Tuple Membership	NL-complete	NL-complete
Walk Membership	P	P
Walk Enumeration	DelayP	Open
Distinct Walk Enumeration	DelayP	Open

Future works:

- See if some P problems can be P-complete or NL-complete.
- Static analysis of the semantics: Containment, Equivalence.
- Applications to query languages: Cypher, GQL.

Thank you!