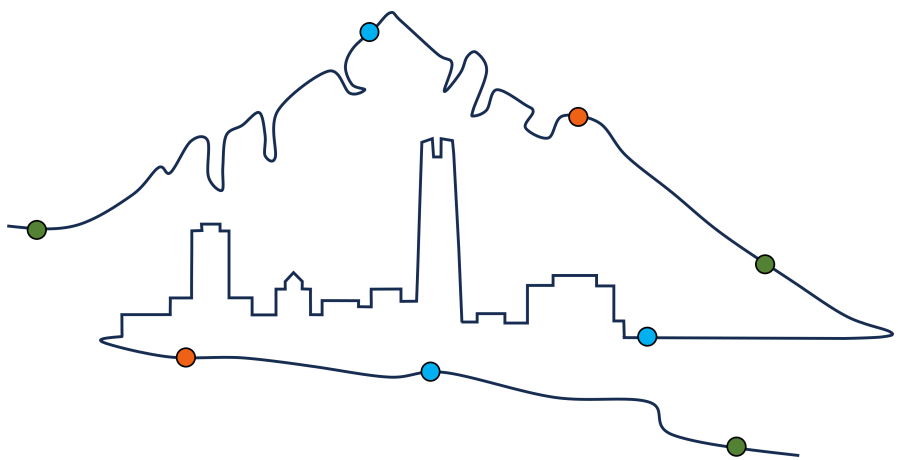
# Distinct Shortest Walk Enumeration for RPQs

Claire DAVID, Nadime FRANCIS and Victor MARSAULT

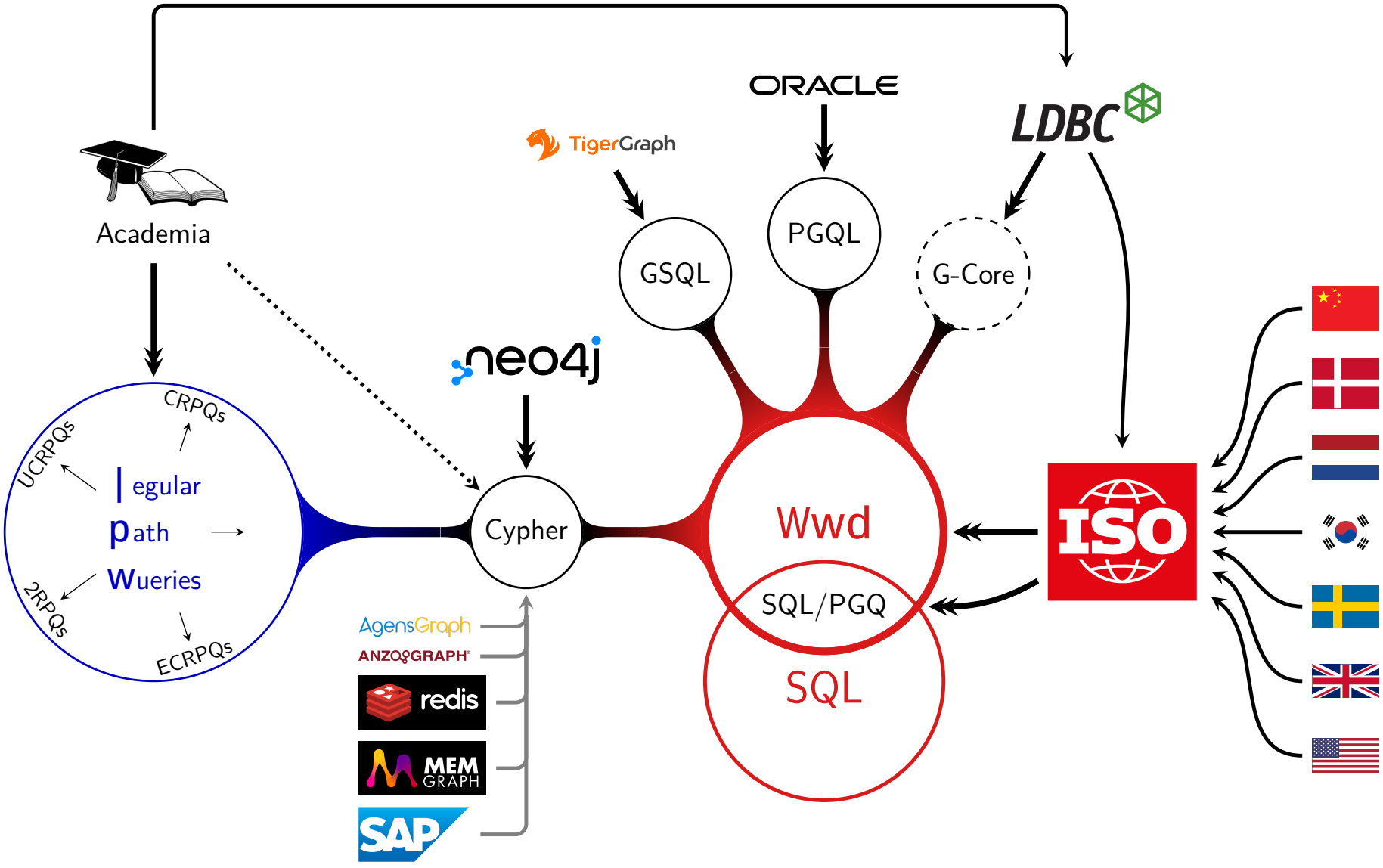
LIGM, Université Gustave Eiffel, CNRS - France



ACM SIGMOD/PODS'24 9-14 June 2024 Santiago, Chile

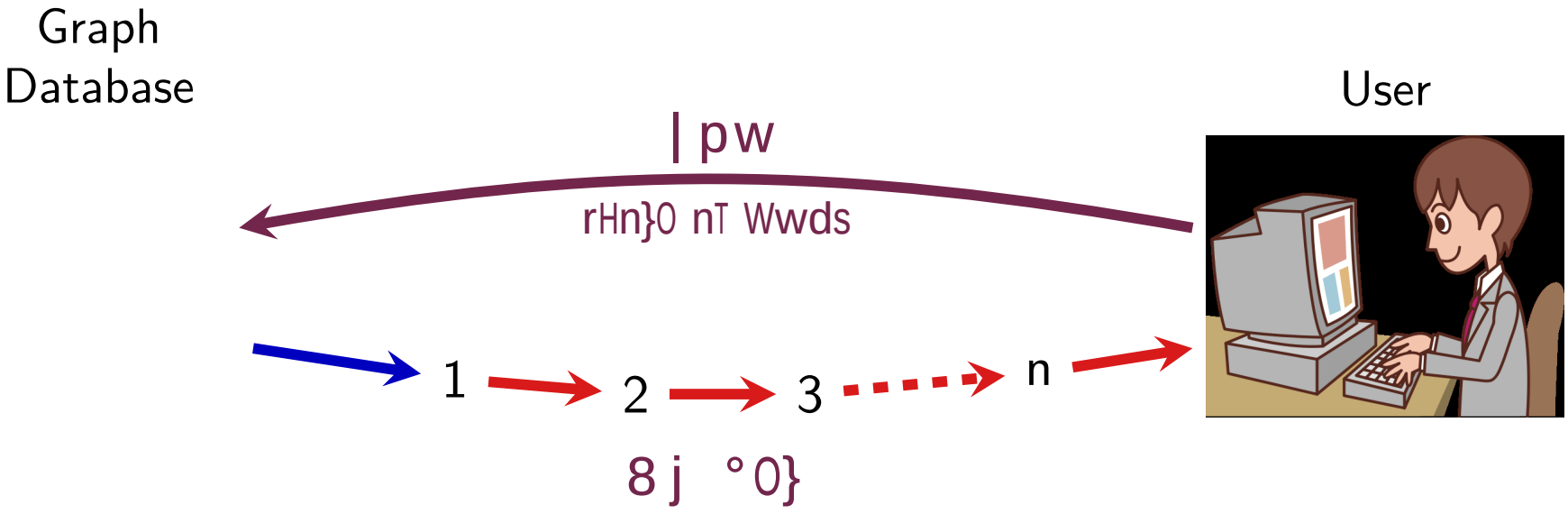
# Ecosystem around GQL





# Our problem





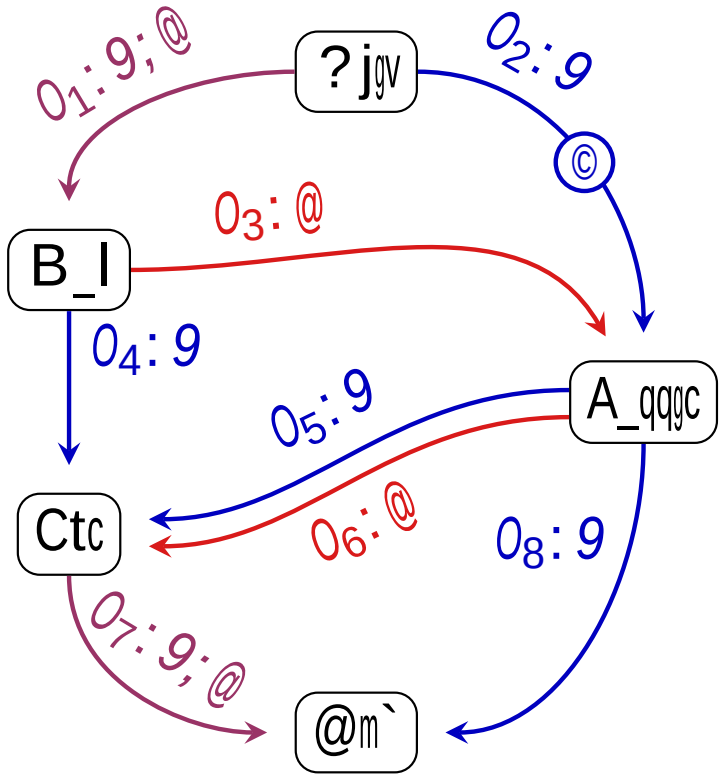

## How to enumerate answers

- Without duplicate
- With good complexity
  - p}0q}nH0 _jX: time before first answer
  - K0e9´: time between consecutive answers
  - Memory Usage

# Data model: Multi-labeled multi-edge graph



- Finite label alphabet: = *f9; @g*
- Vertices: *f ? jgv; @m` ; : : : ; Ctcg*
- Labelled edges: *f01; : : : ; 07g*
  - $0_1$ carries two labels: *9* and *@*
  - $0_2$ carries one label: *9*
  -




Graphs are multi-edge and multi-labeled like GQL data model

- ¯ 9ec = consistent sequence of edges
  $0_1 0_4 0_7$ is a walk $0_1 0_8$ is not a walk
- d9@0e nT 9 °9ec = concatenation of its edge labels
  $0_1 0_4 0_7$ carries four labels: *999 @99 99@ @9@*

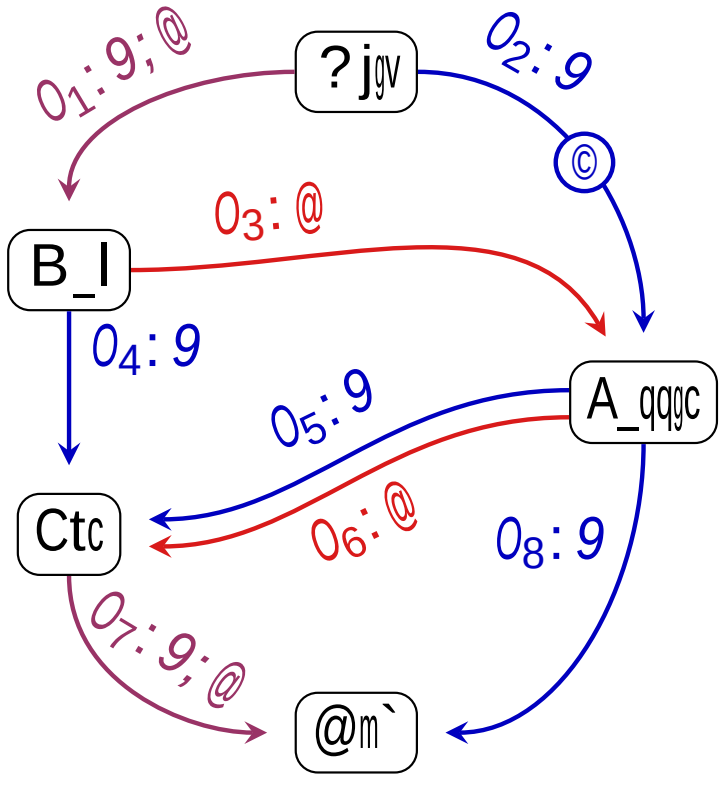
# Query language : RPQ



| | | |
|---|---|---|
| $w ::=$ | $8$ | Atoms |
| | $w \ \ w$ | Concatenation |
| | $w + w$ | Disjunction |
| | $w^*$ | Kleene star |

where $8$ is a label in the graph.

Ex: $w = @^* \ \ 9 \ \ (9 + @)^*$




- g 9§H\ §n $w$ = °9¢c in $K$ with a label that conforms to $w$

$0_1 0_4 0_7$ matches $w$ : it carries $@9@$
$0_1 0_3 0_5 0_7$ matches $w$ : it carries $@@9@$

$0_1 0_4 0_7$ matches $w$ three times and $0_1 0_3 0_5 0_7$ four times
Matches may have different length

# Problem: Distinct Shortest Walk Enumeration



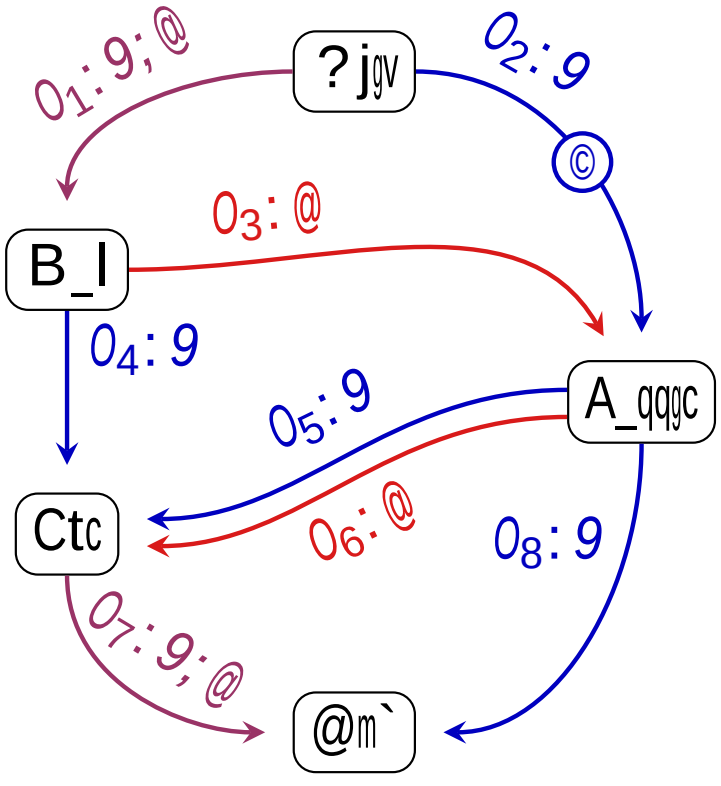
Enumerate all \n}§0 § walks in $K$ from to § matching $w$ °_§\n«§ L«qe_H9§0 .

Ex: = ?jgv;§ = @m`;$w$ = @ 9 (9 + @)

- No matches of length 1 or 2
- $0_1 0_4 0_7$
- $0_1 0_3 0_8$
- $0_2 0_5 0_7$
- $0_2 0_6 0_7$

} Shortest walks : length 3

- $0_1 0_3 0_5 0_7$ has length >3




Output each answer only once

# Previous results



## Martens, Trautner 2018

Distinct enumeration of all shortest walks can be done with

- Polynomial time preprocessing
- Polynomial time delay

Based on Ackerman, Shallit, 2009

## Folklore

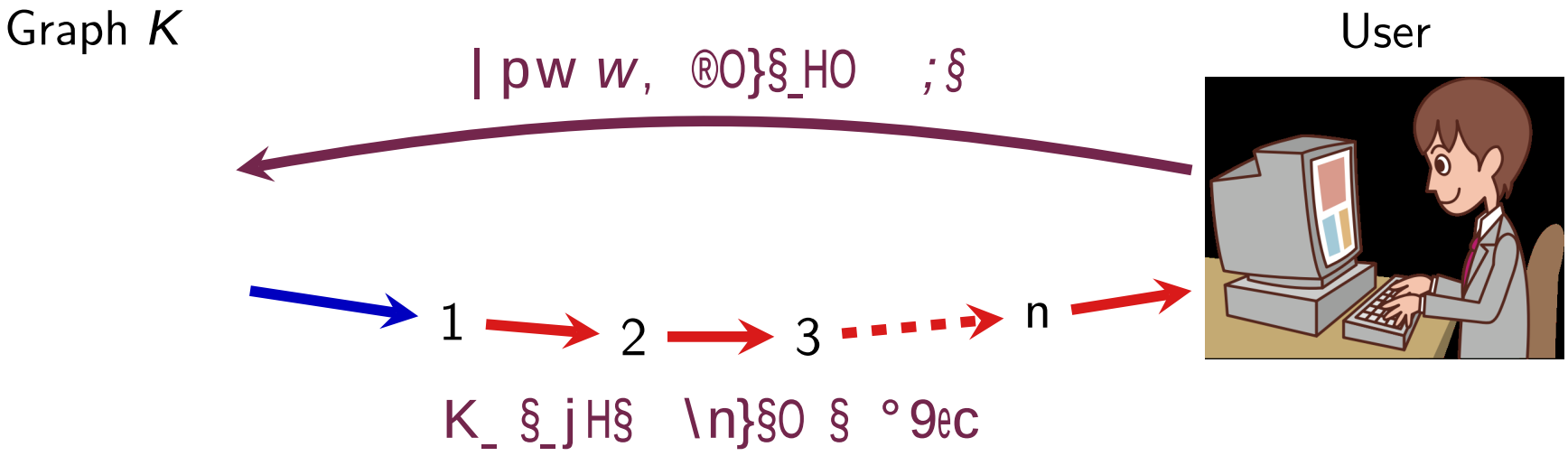
When duplicates cannot occur, enumeration can be done with

- m *jwj* *jKj* preprocessing
- m( ) delay

where is the length of one output.

# Our contribution






An algorithm to enumerate distinct shortest walks with

- m *jwj* *jKj* preprocessing (time before first answer)
- m *jwj* *j j* delay (time between two answers)
- m *jwj* *jKj* memory usage

where is the length of one output.

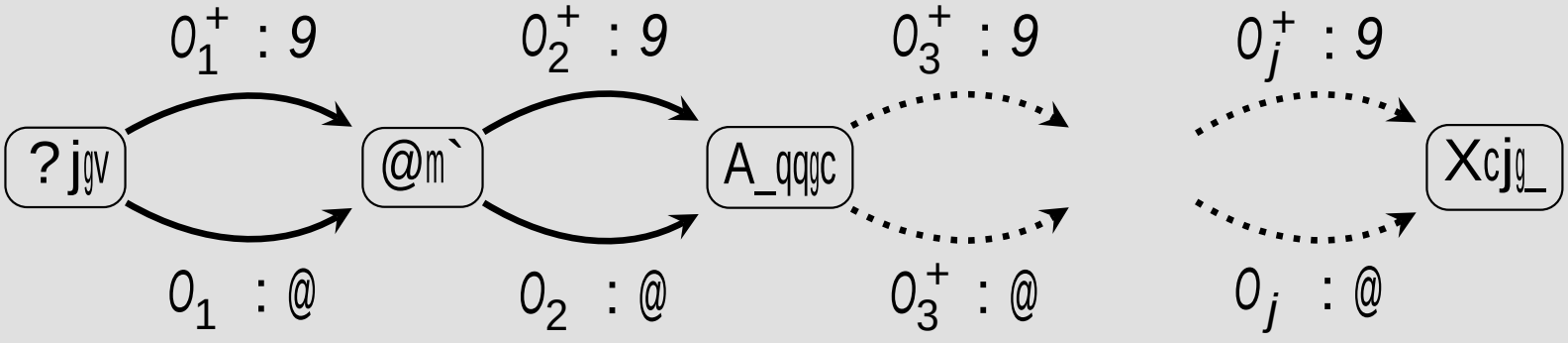
# Exponentially many answers



There are $2^j$ walks from ?jgv to Xcjg_ matching $w$

Q: $(9 + @)$

D:

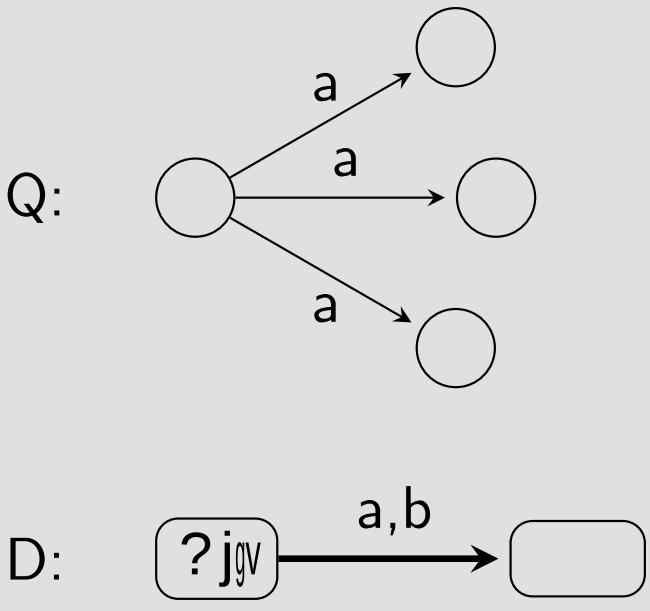



Memory usage must not be linear in the size of output

# Where do duplicates comes from?

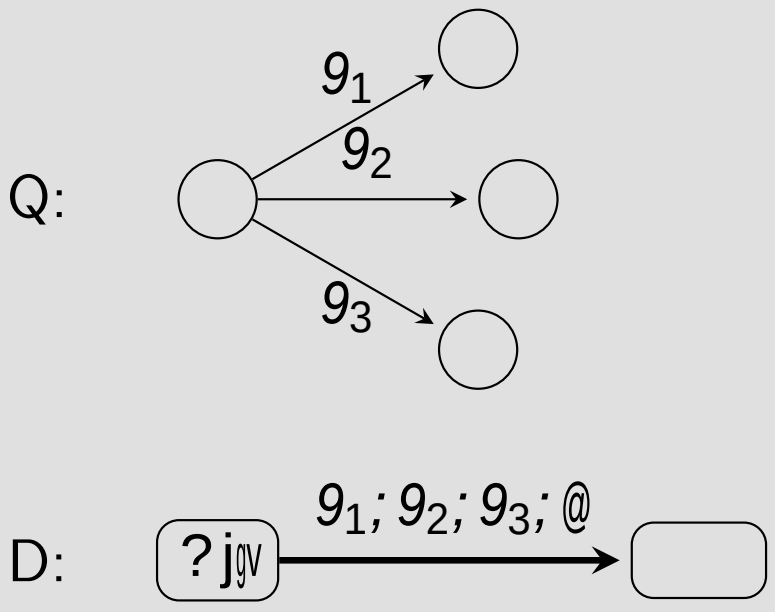


## From the query




( )

## From the data



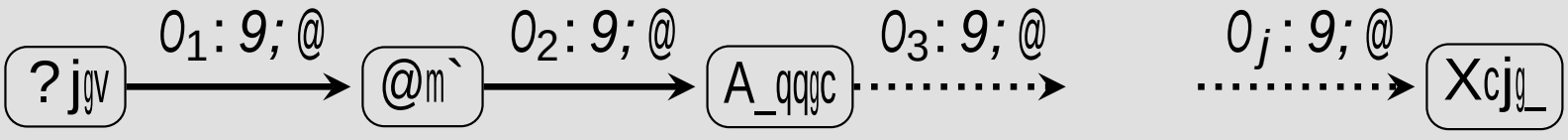

Determinizing the query/automaton does not help

# Exponentially many duplicates



The walk from ?jgv to Xcjg_ matching $w$ has $2^j$ duplicates

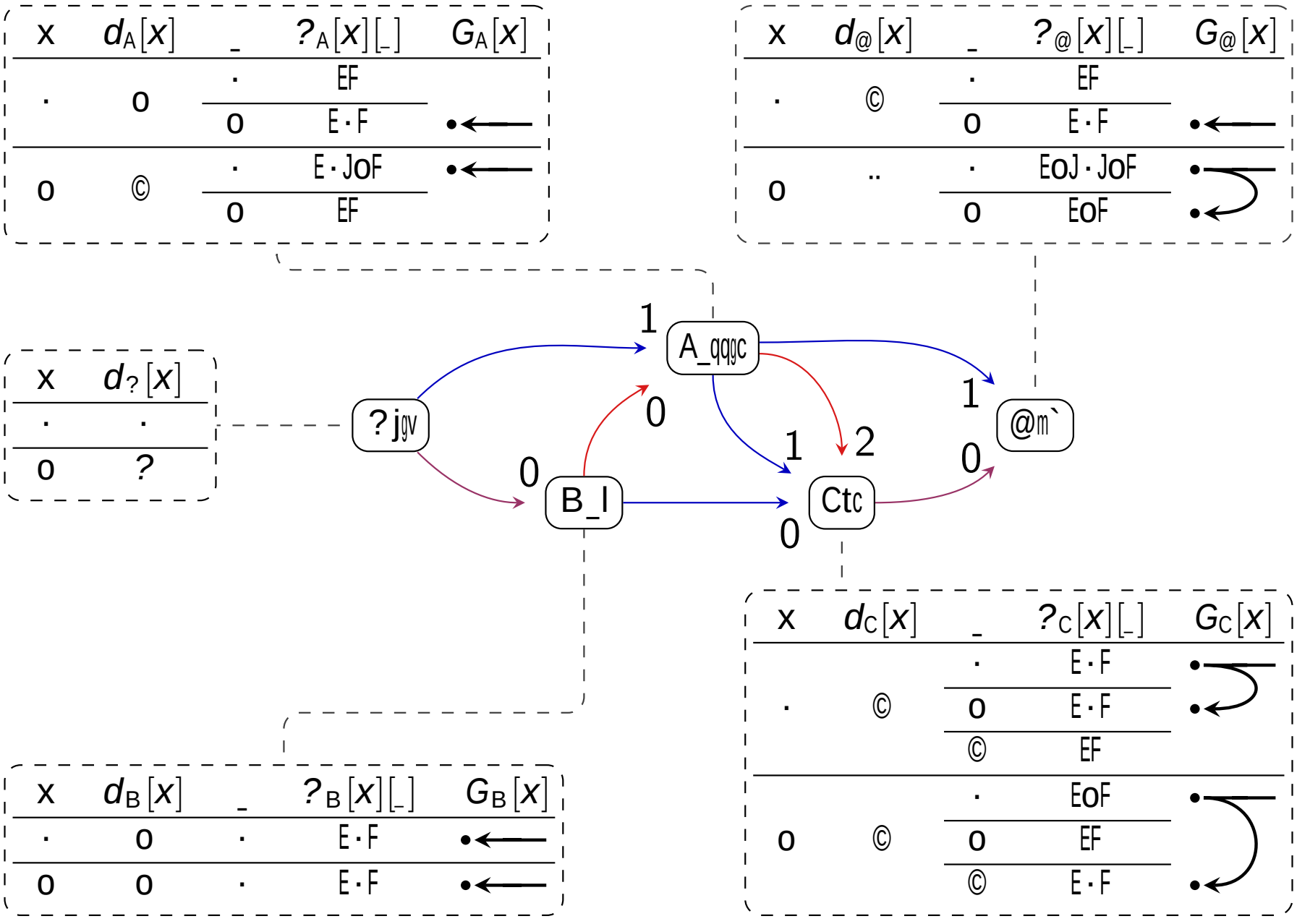
Q: $(9 + @)$

D: 


The algorithm cannot check whether a walk was already output

# Preprocessing : Annotation of $K$





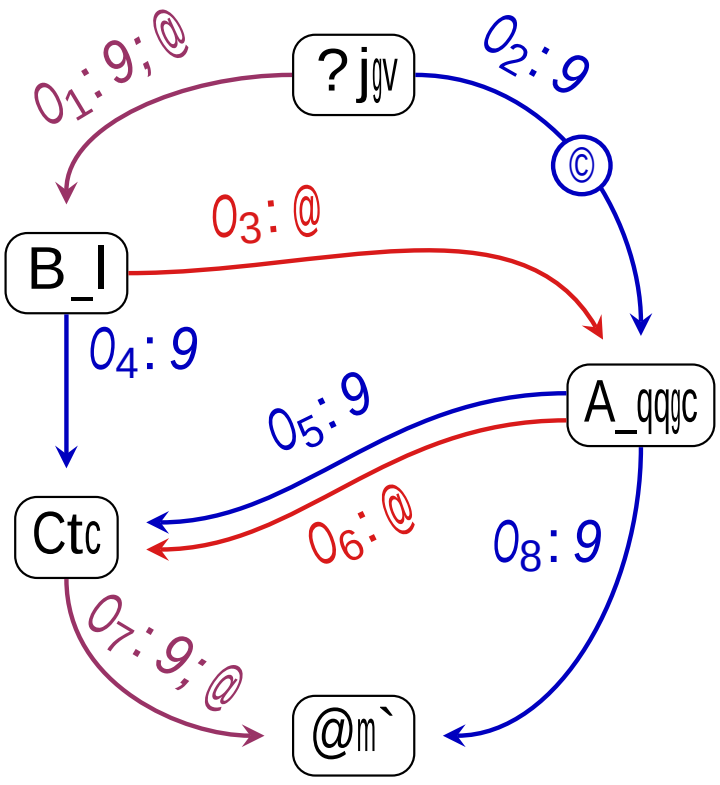
# Enumeration strategy: DFS of the backward tree

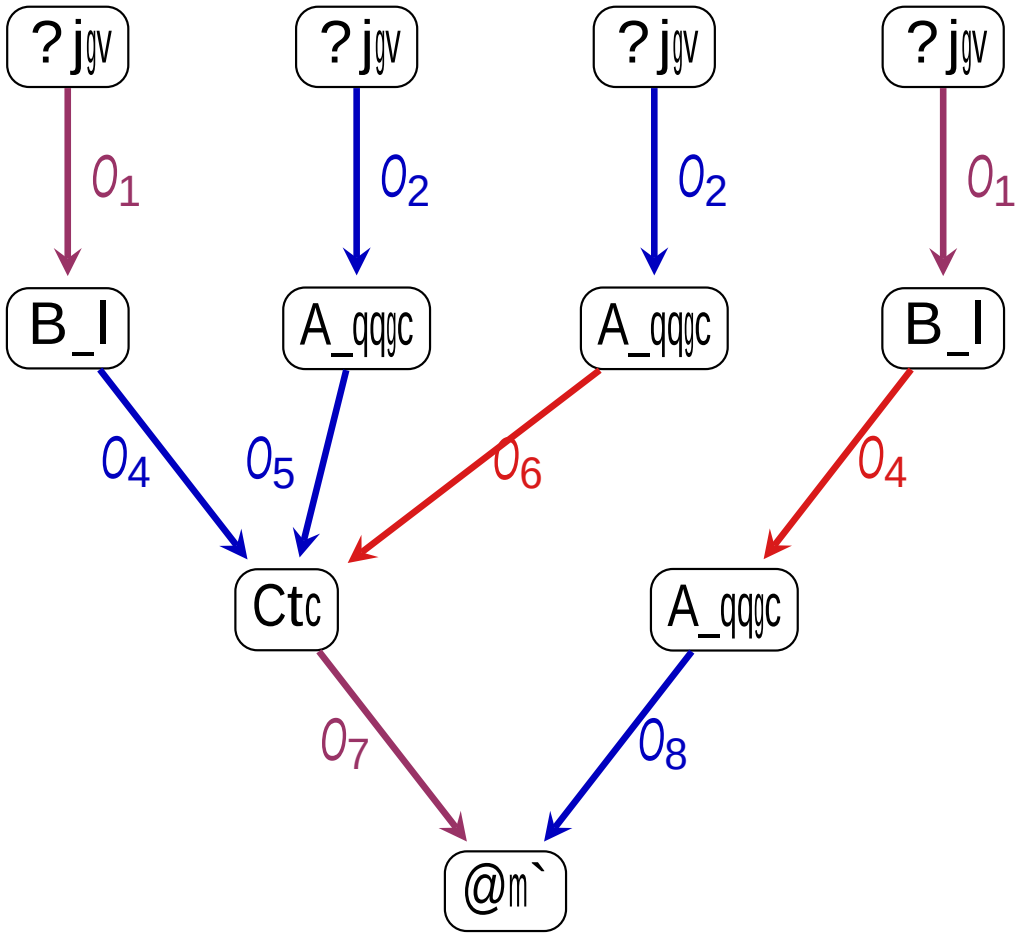


Inputs:

$w$ = @ 9 (9 + @)

= ?jgv § = @m`




Backward tree of answers

r«q §n 0²qnj0j§_9e _j *jKj* 9jL *jwjs*

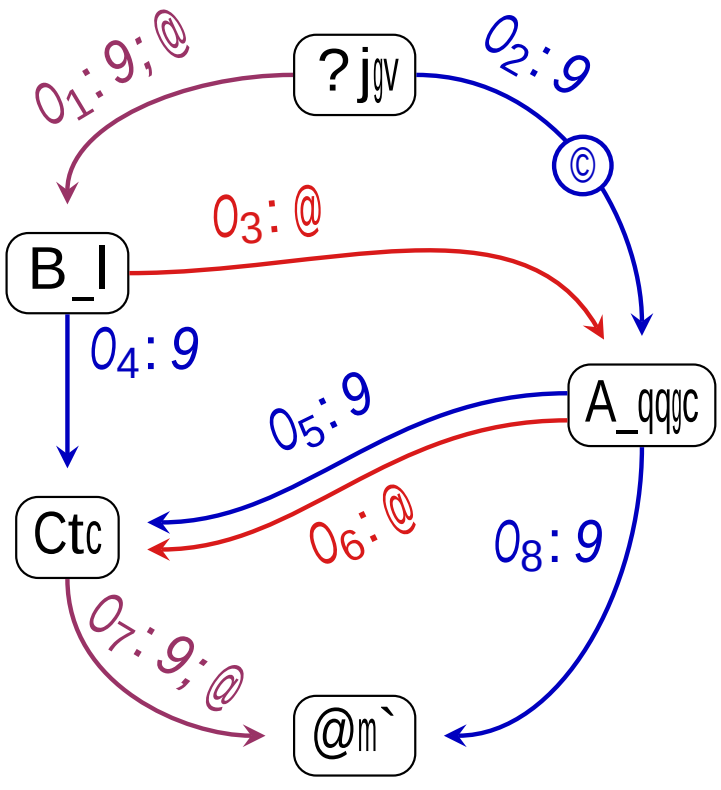
# Enumeration strategy: DFS of the backward tree

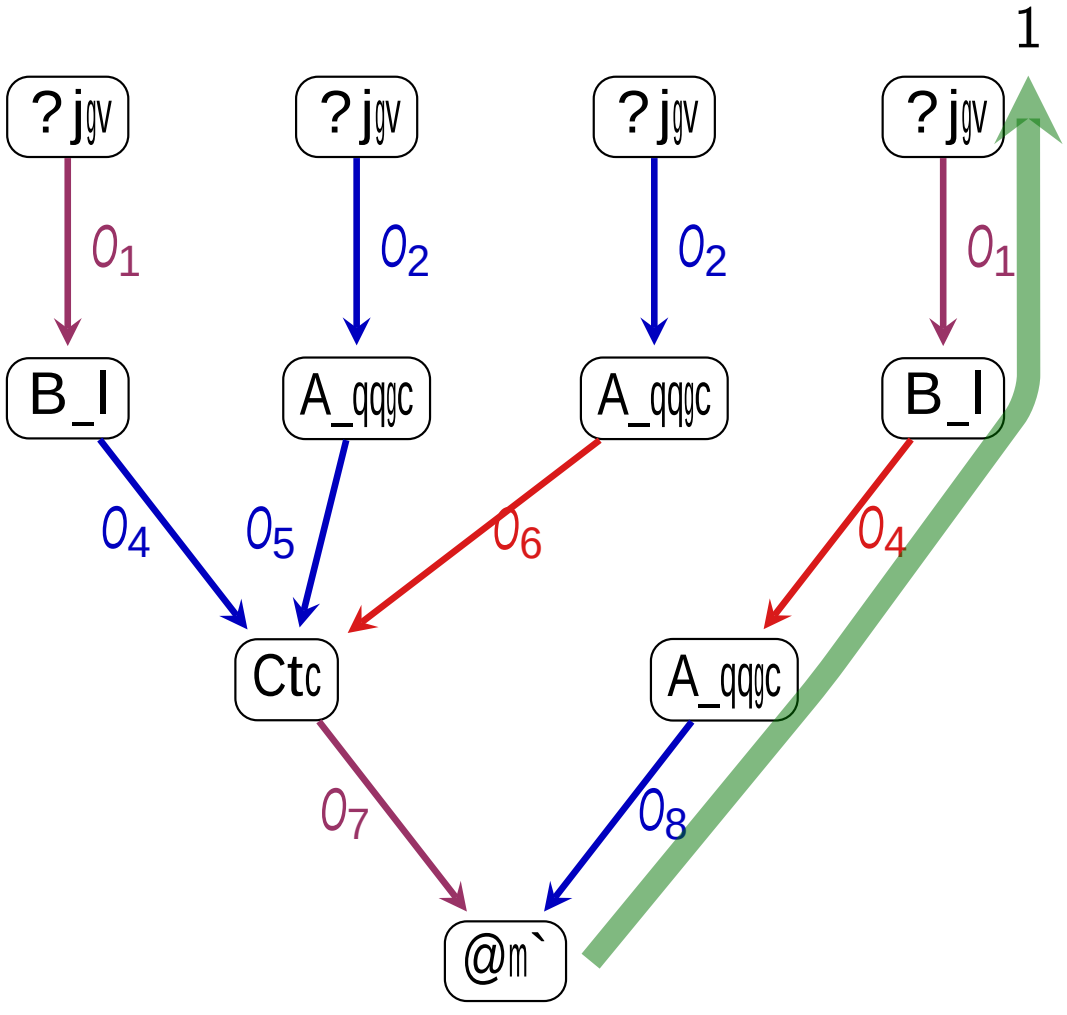


Inputs:

$w$ = @ 9 (9 + @)

= ?jgv § = @m`




Backward tree of answers

r«q §n 0²qnj0j§_9e _j *jKj* 9jL *jwjs*

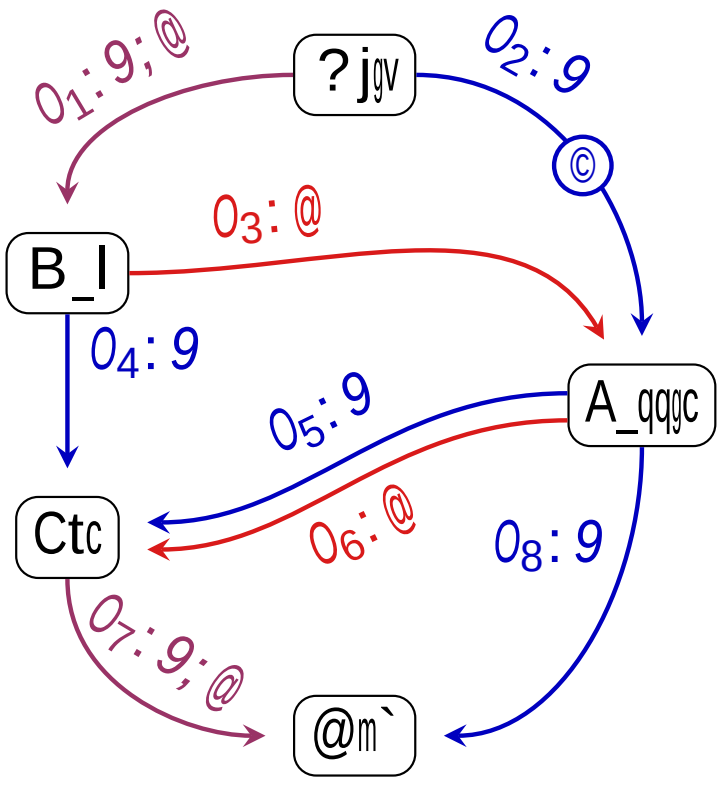
# Enumeration strategy: DFS of the backward tree

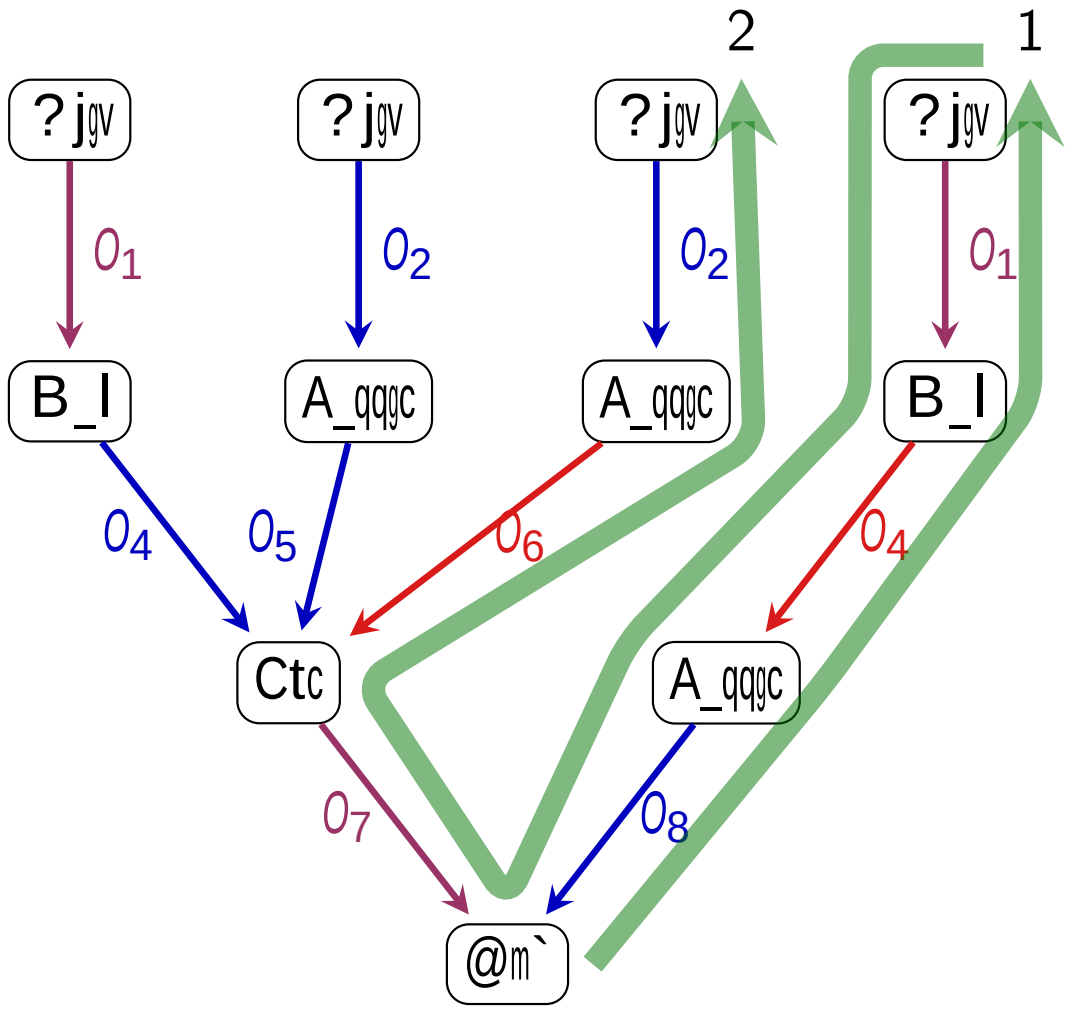


Inputs:

$w$ = @ 9 (9 + @)

= ?jgv § = @m`




Backward tree of answers

r«q §n 0²qnj0j§_9e _j *jKj* 9jL *jwjs*

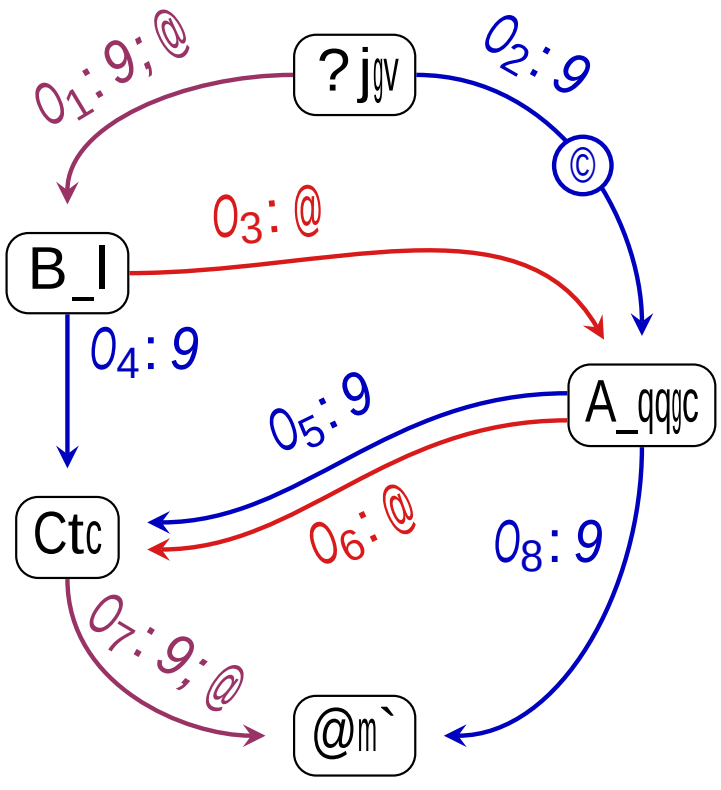
# Enumeration strategy: DFS of the backward tree

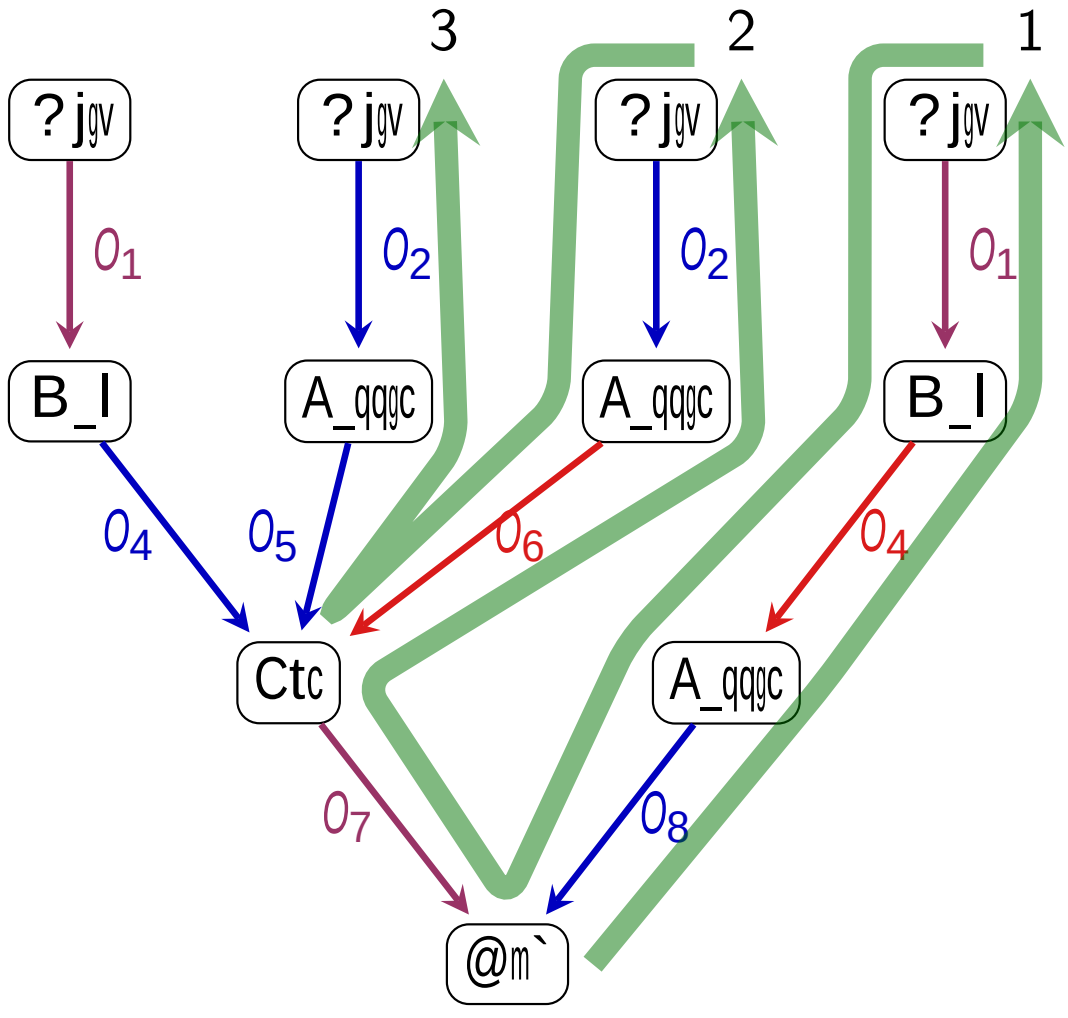


Inputs:

$w$ = @ 9 (9 + @)

= ?jgv § = @m`




Backward tree of answers

r«q §n 0²qnj0j§_9e _j *jKj* 9jL *jwjs*

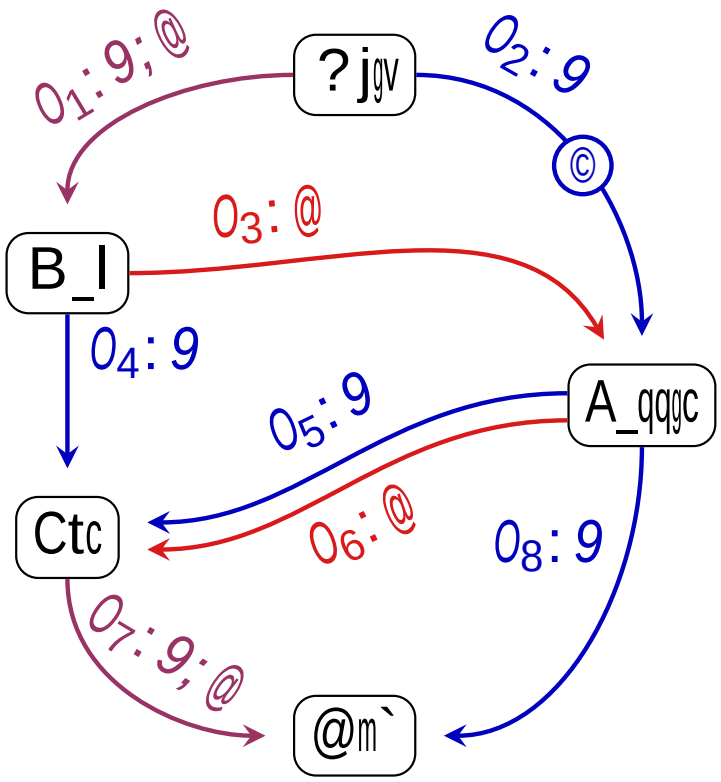
# Enumeration strategy: DFS of the backward tree

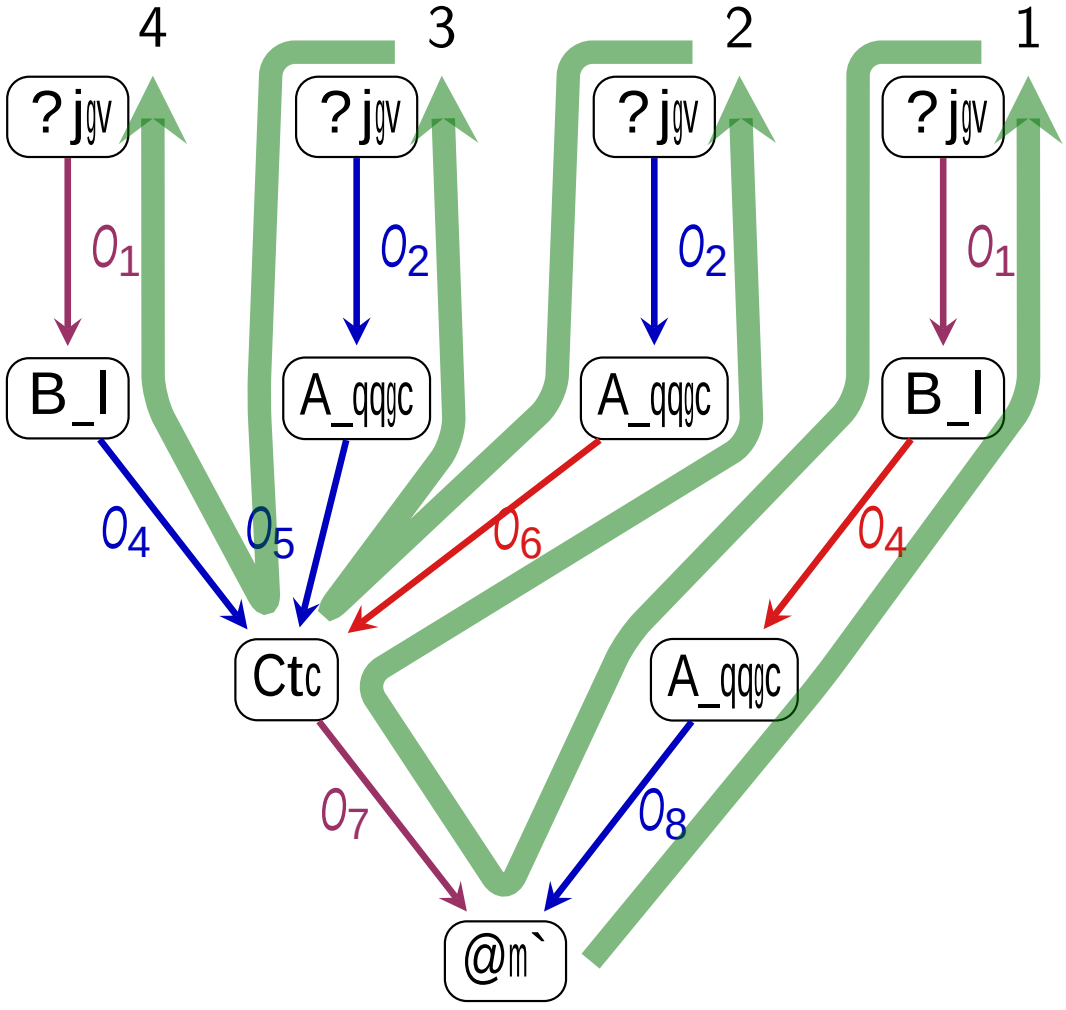


Inputs:

$w$ = @ 9 (9 + @)

= ?jgv § = @m`




Backward tree of answers

r«q §n 0²qnj0j§_9e _j *jKj* 9jL *jwjs*

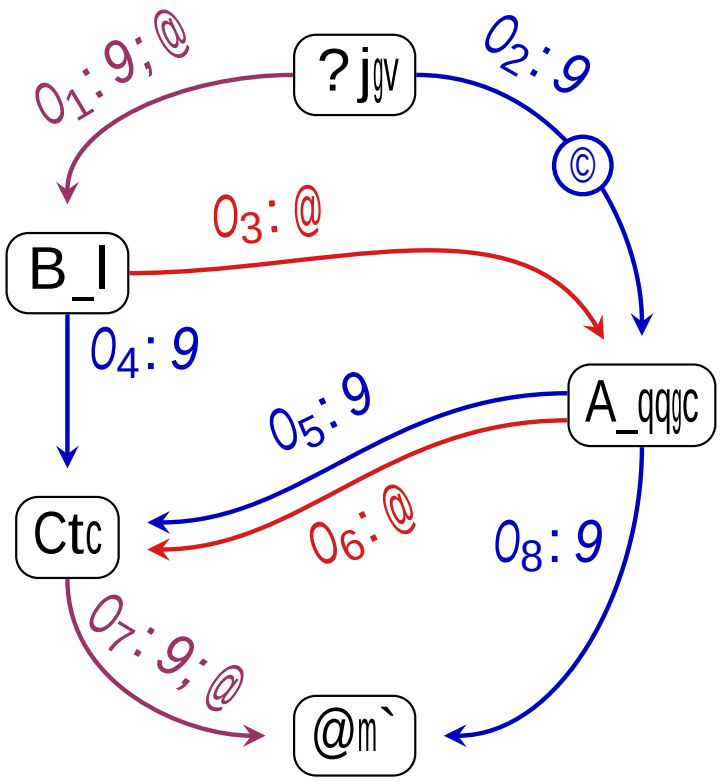
# Enumeration strategy: DFS of the backward tree

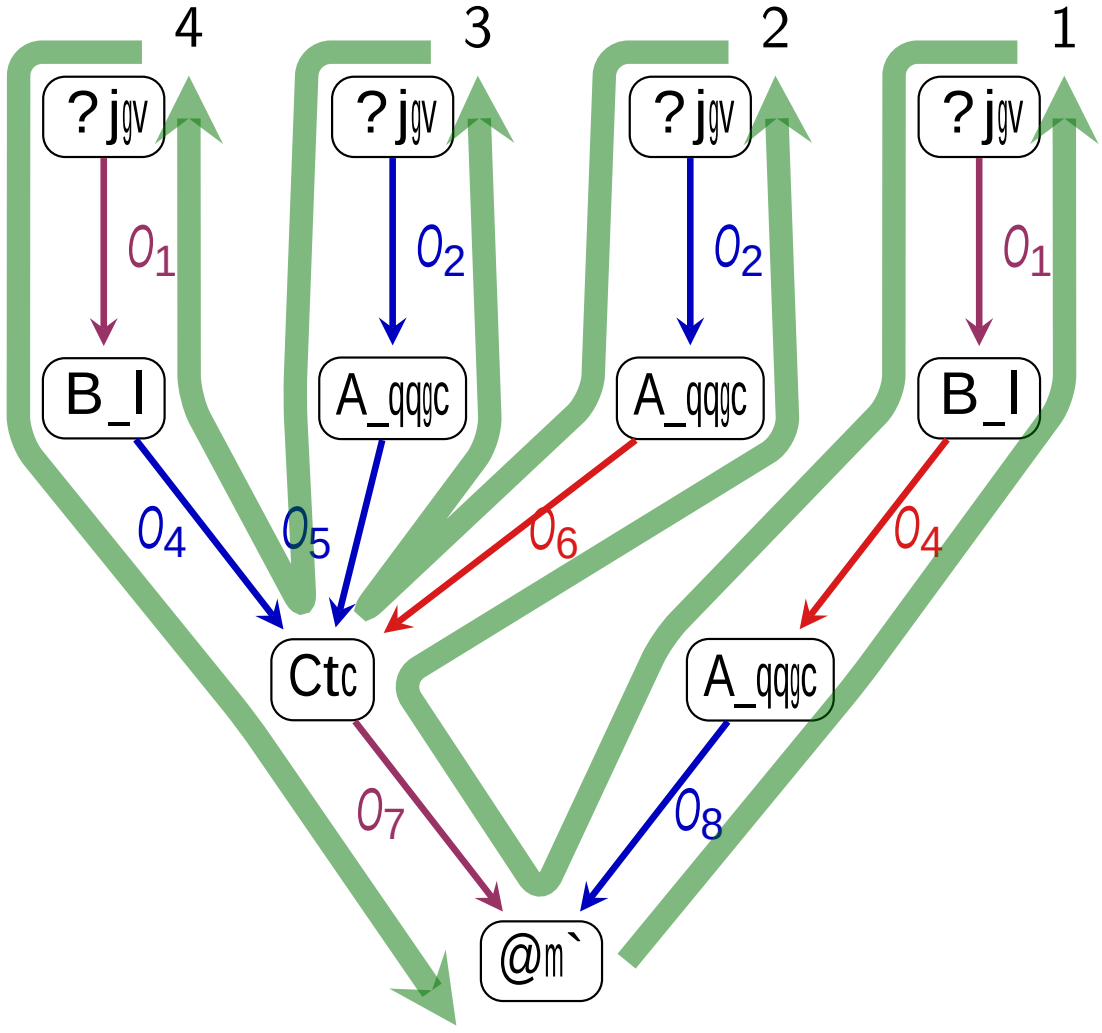


Inputs:

$w$ = @ 9 (9 + @)

= ?jgv § = @m`




Backward tree of answers

r«q §n 0²qnj0j§_9e _j *jKj* 9jL *jwjs*

# Our algorithm in one slide



## Preprocessing phase

- Annotate $K$ using a BFS of product graph $K \quad w$
- Reindex the annotation to encode the backward tree

## Enumeration phase

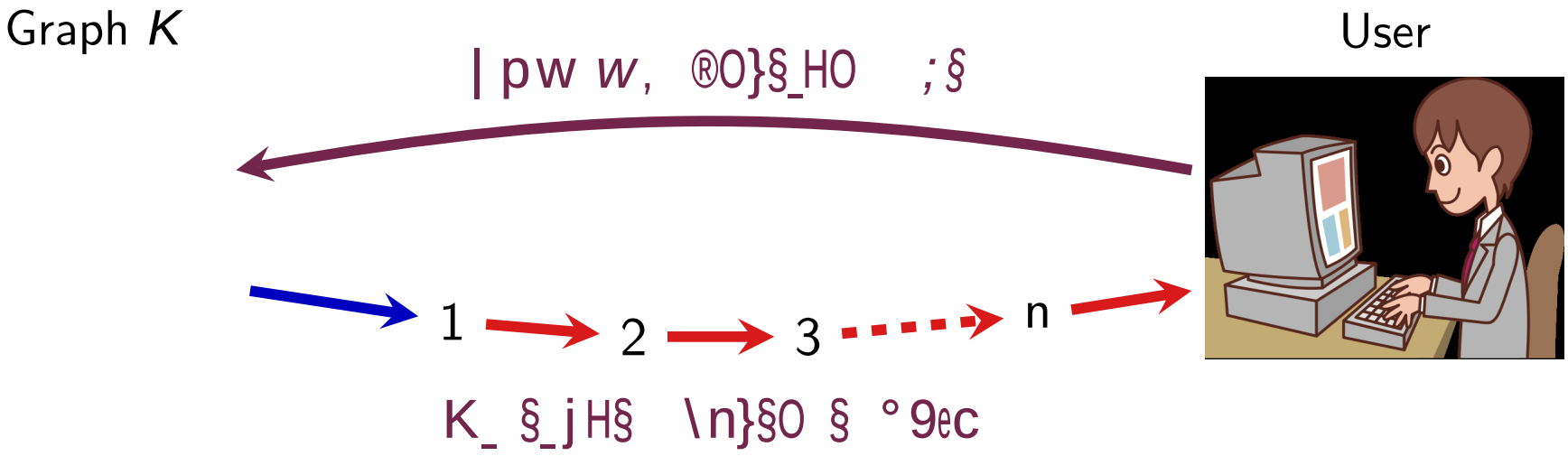
- DFS of the backward tree, computed on-the-fly
- No dead-ends
- Find next children independently of local topology
- Annotation is almost read-only

## Carefull use of classical data structures

- "Sorting" in linear time via bucket-sort
- Shortcuts in the data structure via LinkedLists
- N-ary "zipper" to merge sorted lists

# Our contribution





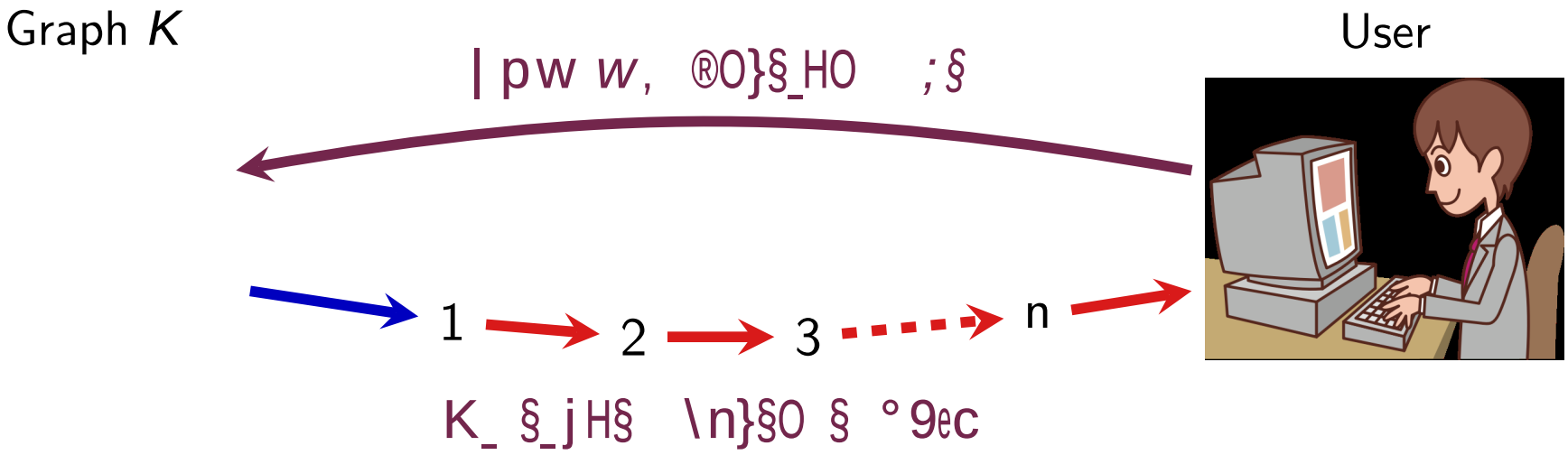

An algorithm to enumerate distinct shortest walks with

- m *jwj* *jKj* preprocessing (time before first answer)
- m *jwj* *j j* delay (time between two answers)
- m *jwj* *jKj* memory usage

where is the length of one output.

# Some perspectives






- Reducing overhead when nondeterminism does not occur
- Add other GQL features

- Are data structures well-behaved ?
- Gather information about nondeterminism in real-life settings