Compact Answers to Temporal Path Queries

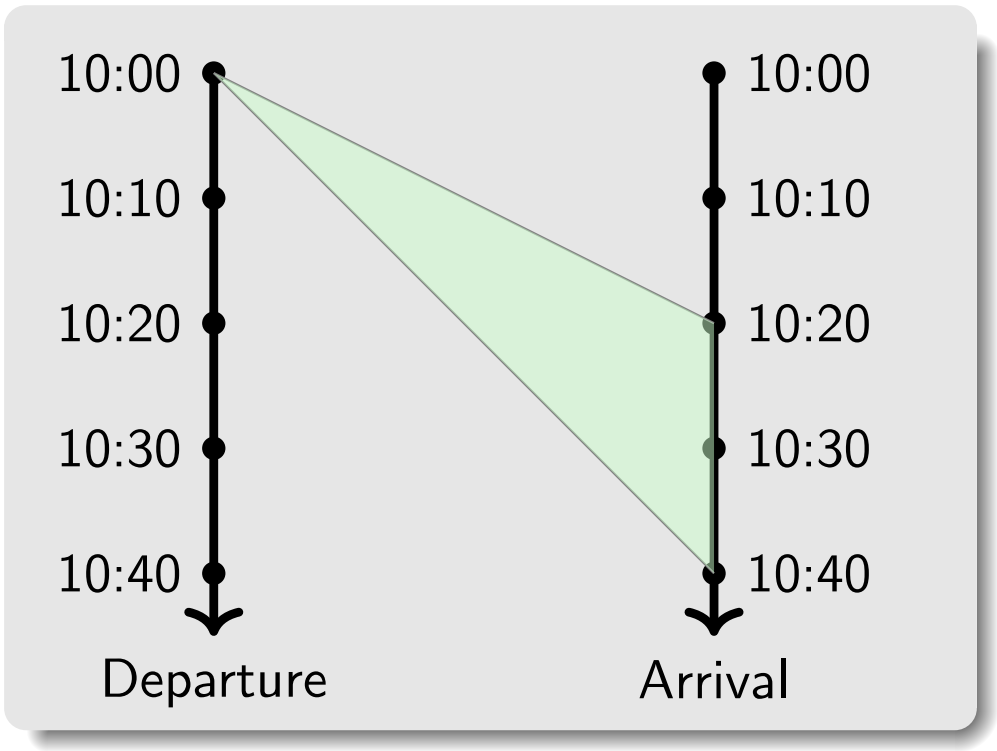
Muhammad Adnan Diego Calvanese Julien Corman
Anton Dignös Werner Nutt Ognjen Savković

Free University of Bozen-Bolzano

5 November 2025

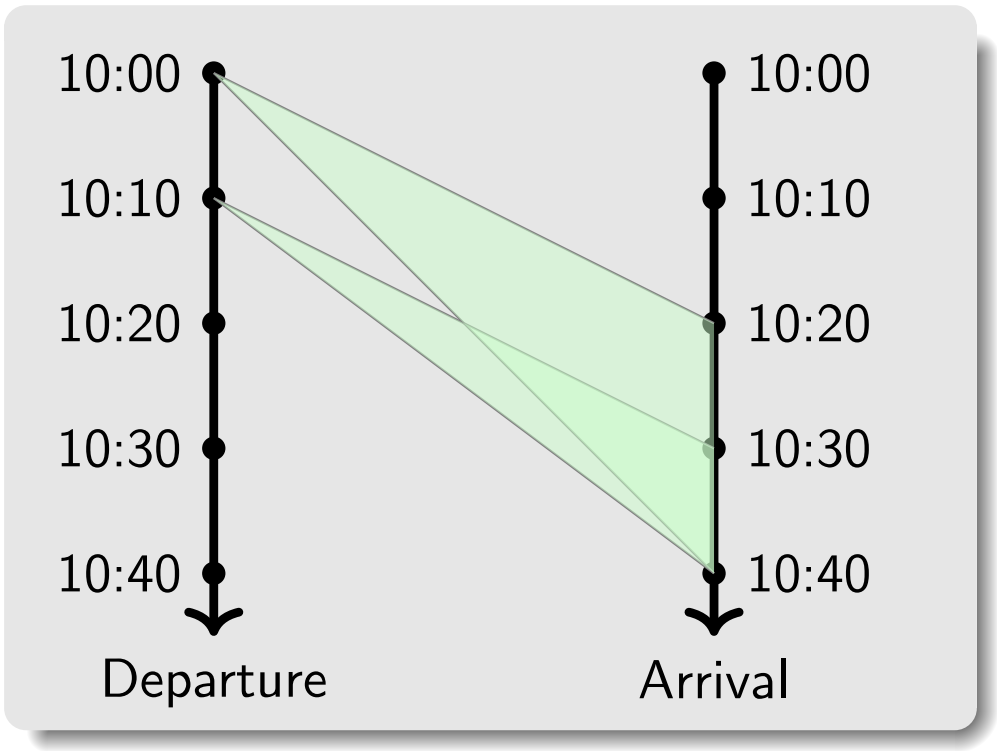
The time span between two events may be uncertain

- A road trip
  with **uncertain traffic**,



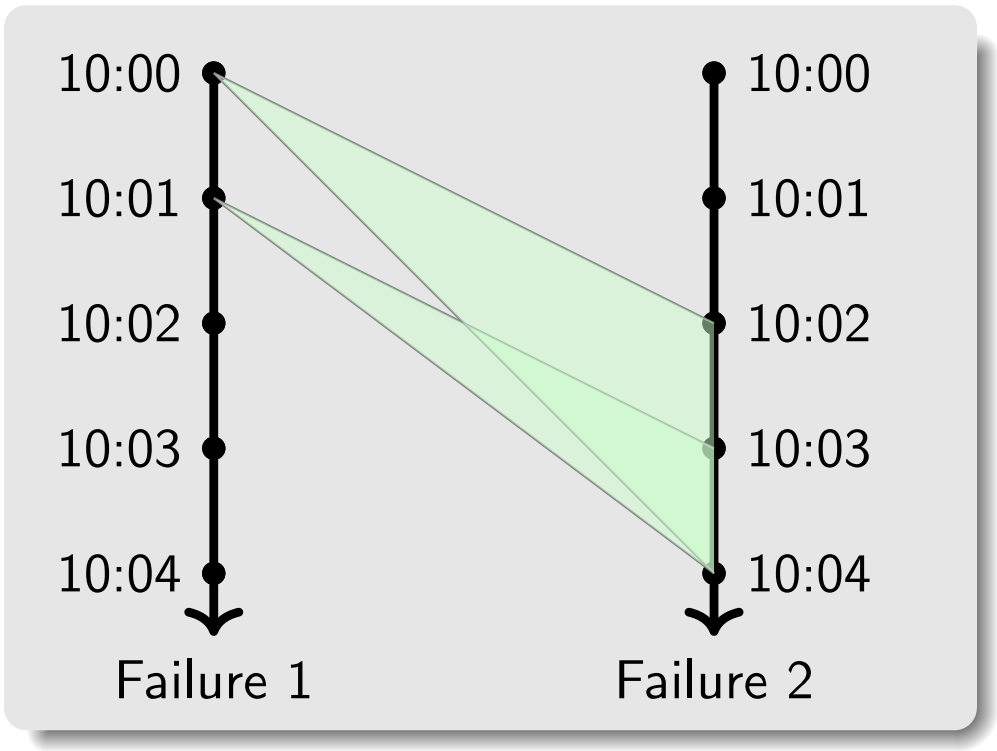
The time span between two events may be uncertain

- A road trip
  with **uncertain traffic**,



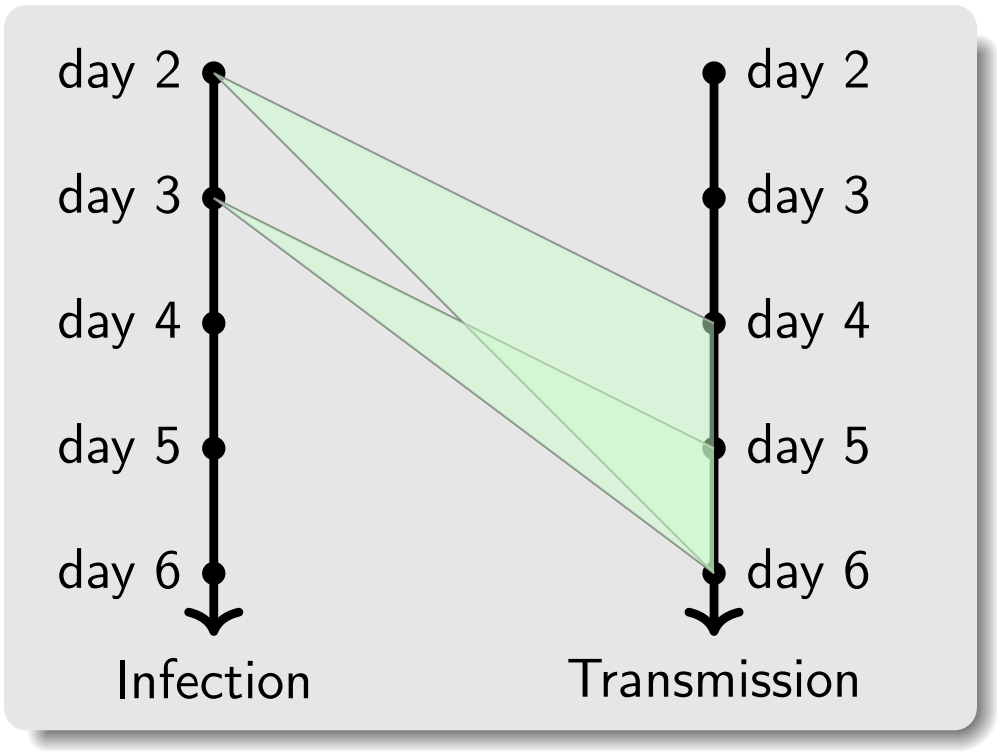
The time span between two events may be uncertain

- A road trip
  with **uncertain traffic**, or
- malfunctions of
  two components,



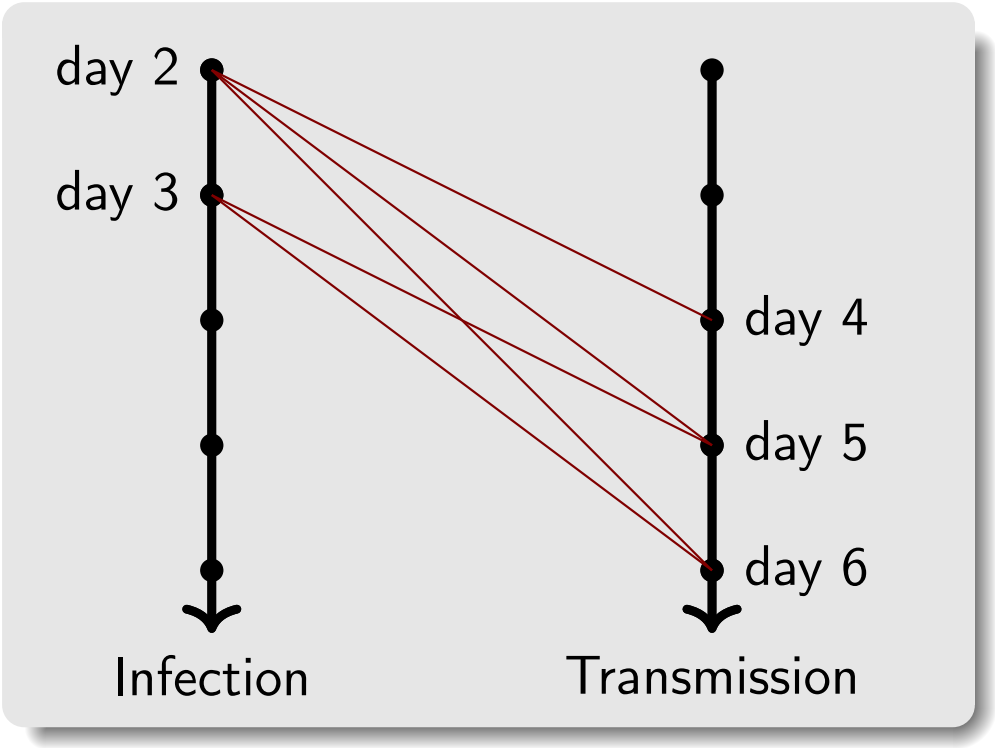
The time span between two events may be uncertain

- A road trip with **uncertain traffic**, or
- malfunctions of **two components**, or
- the **spread** of a virus, or
- ...



This information can be viewed as a relation over time points

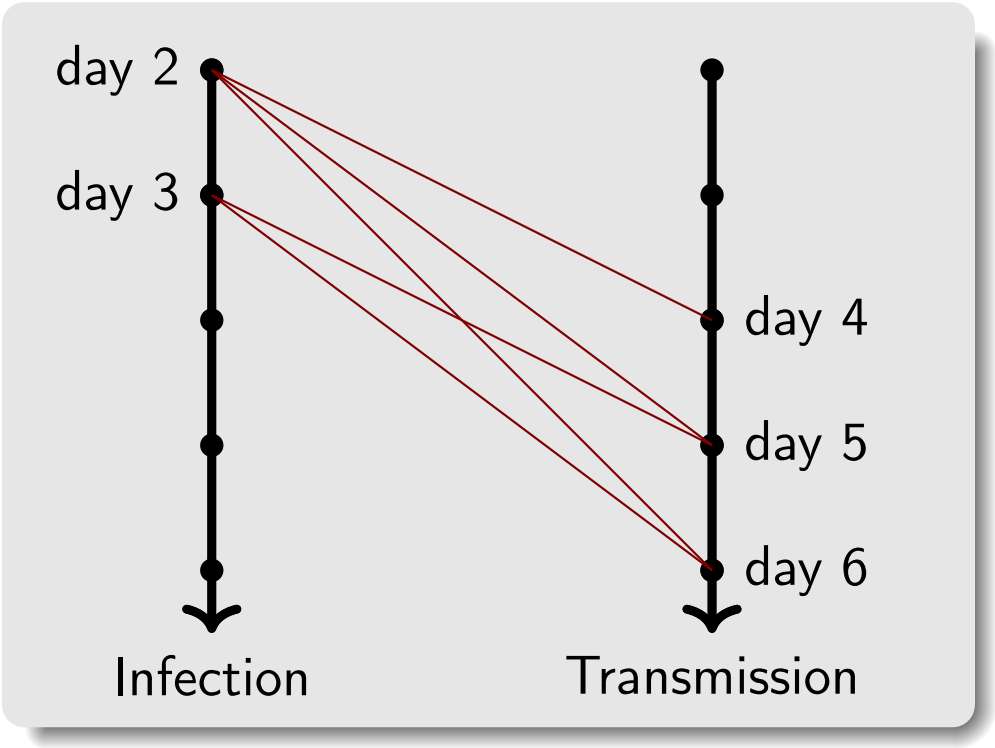
Infection	Transmission
2	4
2	5
2	6
3	5
3	6



This information can be viewed as a relation over time points

Infection	Transmission
2	4
2	5
2	6
3	5
3	6

- We call *temporal relation* a binary relation over time points.



Temporal relations may be associated to data

Person 1	Person 2	Infection	Transmission
Alice	Bob	2	4
		2	5
		2	6
		3	5
		3	6

Temporal relations may be associated to data

Person 1	Person 2	Infection	Transmission
Alice	Bob	2	4
		2	5
		2	6
		3	5
		3	6
Alice	Carol	2	4
		2	5
		3	5
Bob	Carol	4	6

This format generalizes standard temporal data

- We associate a temporal **relation** to each tuple of data.

A	B	Time_1	Time_2
a	b	t_1	t_2
		t_1	t_3
		t_2	t_3
		...	...
a	c	t_1	t_2
		...	...
b	c	t_1	t_4
		...	...

This format generalizes standard temporal data

- We associate a temporal **relation** to each tuple of data.

A	B	Time_1	Time_2
a	b	t_1	t_2
		t_1	t_3
		t_2	t_3
		...	...
a	c	t_1	t_2
		...	...
b	c	t_1	t_4
		...	...

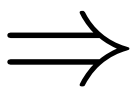
- Temporal databases associate a **set** of time points to each tuple of data.

A	B	Time
a	b	t_1
		t_2
		t_3
		...
a	c	t_1
		...
b	c	t_2
		...

Temporal databases manipulate intervals rather than time points

- Consecutive time points are traditionally **grouped into intervals**.

A	B	Time
a	b	3
		4
		5
		8
		9
c	d	5
		6

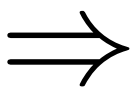


A	B	Time
a	b	$[3, 5]$
		$[8, 9]$
c	d	$[5, 6]$

Temporal databases manipulate intervals rather than time points

- Consecutive time points are traditionally **grouped into intervals**.
- Manipulating time points directly would be:
  - **prohibitive** over discrete time (exponential blowup),

A	B	Time
a	b	3
		4
		5
		8
		9
c	d	5
		6

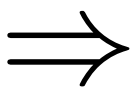


A	B	Time
a	b	$[3,5]$ $[8,9]$
c	d	$[5,6]$

Temporal databases manipulate intervals rather than time points

- Consecutive time points are traditionally **grouped into intervals**.
- Manipulating time points directly would be:
  - **prohibitive** over discrete time (exponential blowup),
  - **impossible** over dense time.

A	B	Time
a	b	3 ... 8 ...
c	d	5 ...



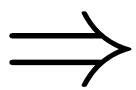
A	B	Time
a	b	$[3,5)$ $[8,9)$
c	d	$[5,6)$

Temporal query engines produce compact answers

- Temporal query engines support operations like
  - temporal join,
  - temporal union,
  - etc.
- These are performed by modifying **interval boundaries**.
- The number of answers is **independent** of the size of the input intervals.

How to represent temporal relations in a compact way?

A	B	Time_1	Time_2
Alice	Bob	2	4
		2	5
		2	6
		3	5
		3	6
Alice	Carol	2	4
		2	5
		3	5
Bob	Carol	4	6

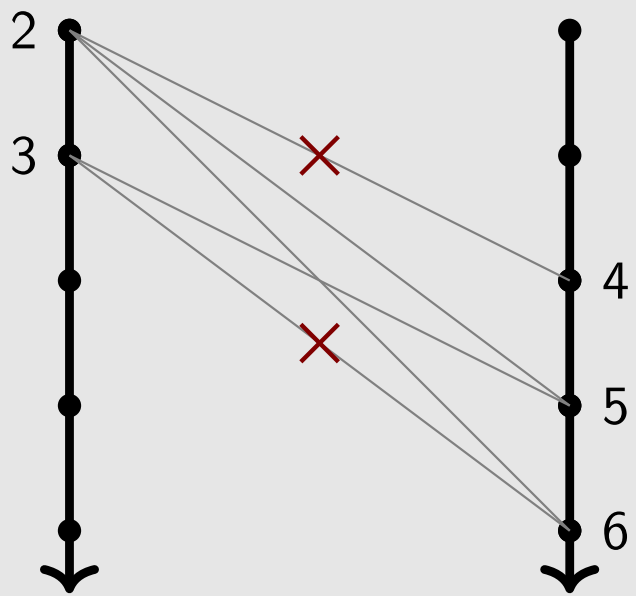


Can we keep answers compact when we query temporal relations?

- Natural operations over temporal **relations** include:

Can we keep answers compact when we query temporal relations?

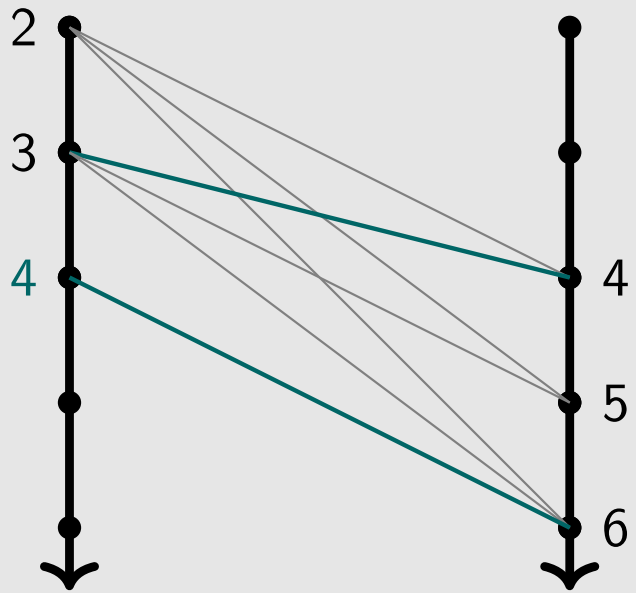
- Natural operations over temporal **relations** include:
  - **Filtering** a relation,



~~2~~	~~4~~
2	5
2	6
3	5
~~3~~	~~6~~

Can we keep answers compact when we query temporal relations?

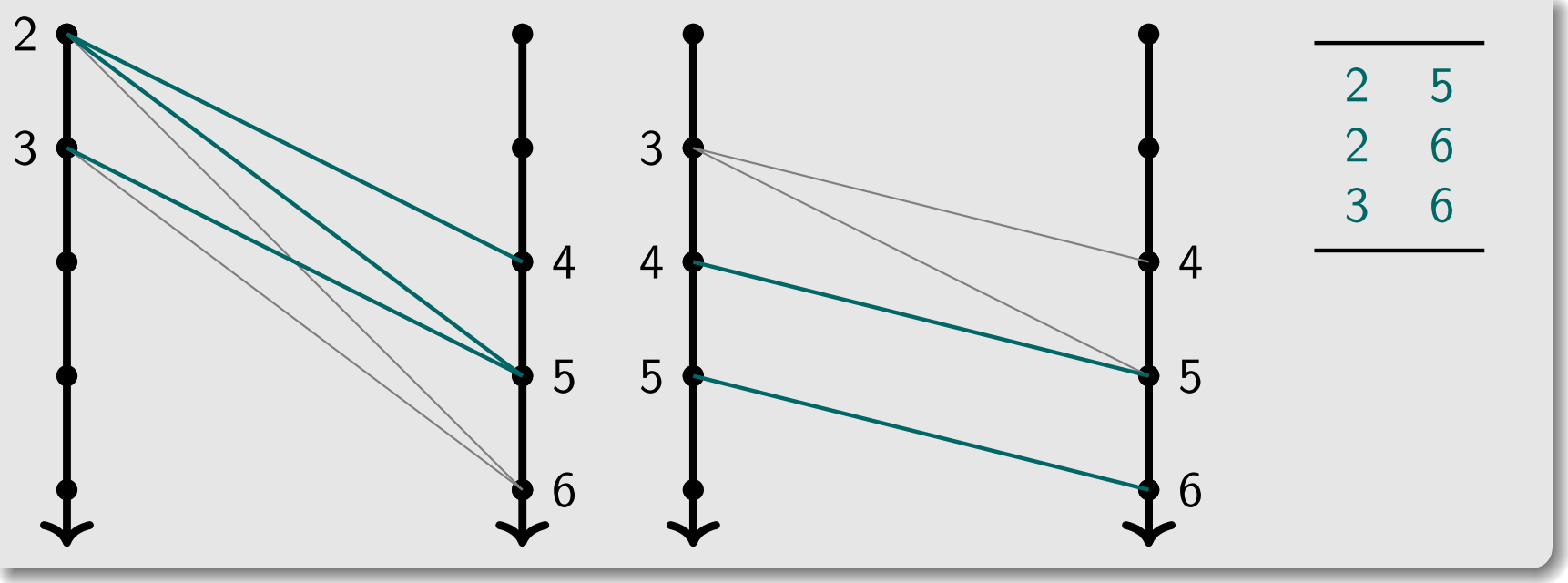
- Natural operations over temporal **relations** include:
  - **Filtering** a relation,
  - **Extending** a relation,



2	4
2	5
2	6
3	4
3	5
3	6
4	6

Can we keep answers compact when we query temporal relations?

- Natural operations over temporal **relations** include:
  - **Filtering** a relation,
  - **Extending** a relation,
  - **Composing** two relations.



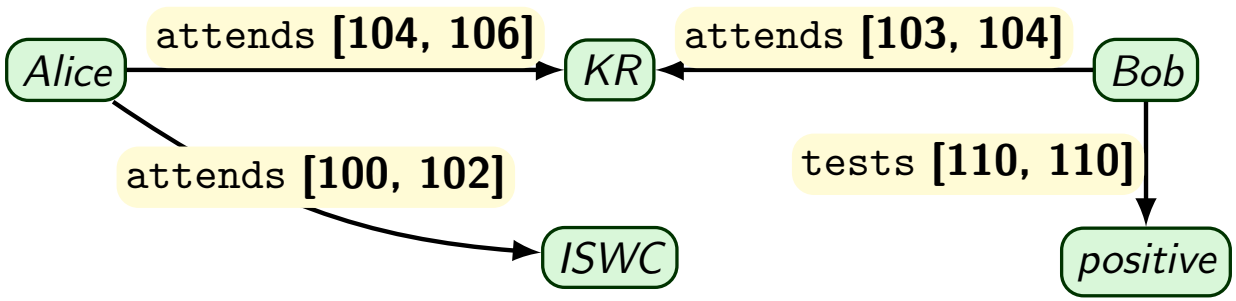
Can we keep answers compact when we query temporal relations?

- Natural operations over temporal **relations** include:
  - **Filtering** a relation,
  - **Extending** a relation,
  - **Composing** two relations.
- **Querying** temporal relations (rather than sets of time points) has **barely been explored**.
- Keeping these relations compact during query evaluation is **non-trivial**.

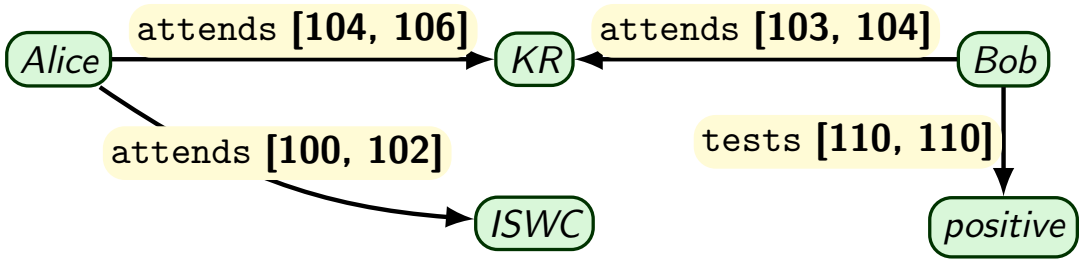
We study this problem for path queries

- We study compact query answering for:
  - *Temporal Regular Path Queries (TRPQs)*,
  - evaluated over *Temporal Knowledge Graphs (TKGs)*.
- Both were introduced in [Arenas et al., 2022].
- TRPQs **output temporal relations**.
- But their compact representation was **left open** in [Arenas et al., 2022].

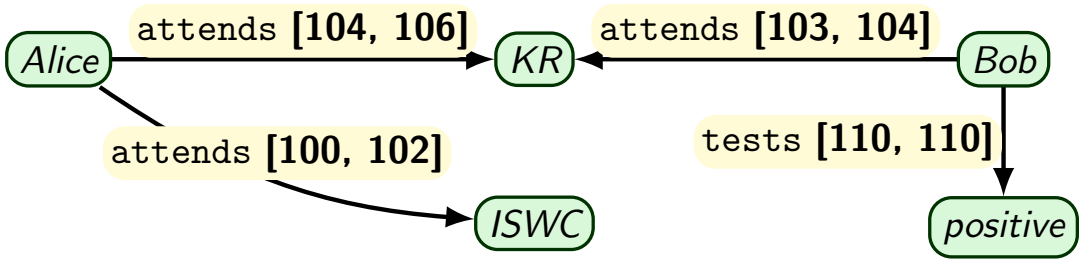
A TKG labels edges with intervals



A TRPQ navigates in a graph, but also in time



A TRPQ navigates in a graph, but also in time



Query

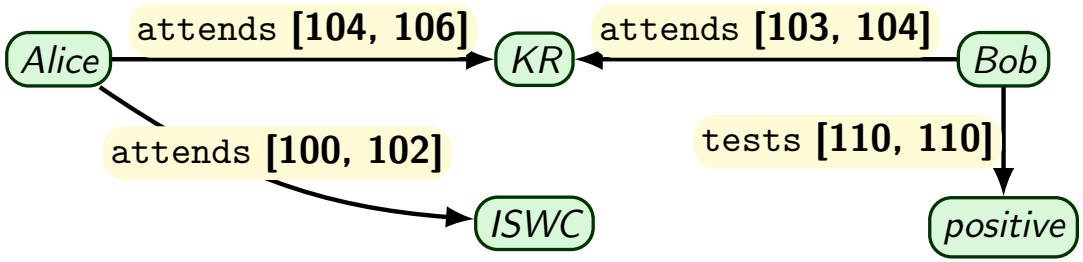
```
attends
```

- "Retrieve all attendees"

Answers

Node_1	Node_2	Time_1	Time_2
Alice	ISWC	100	100
		101	101
		102	102
Alice	KR	104	104
		105	105
		106	106
Bob	KR	103	103
		104	104

A TRPQ navigates in a graph, but also in time



Query

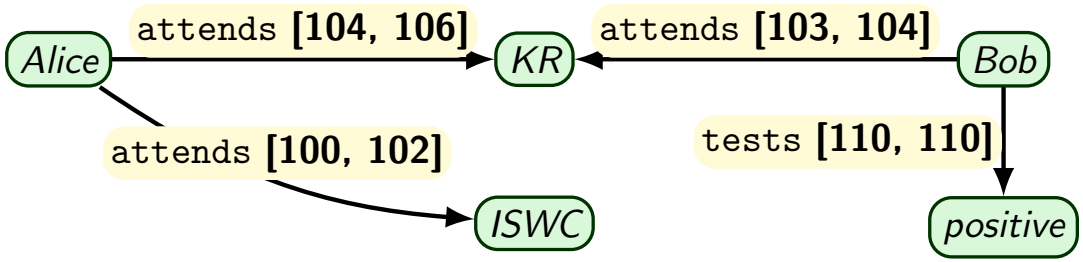
$\texttt{attends}/(= ISWC)$

- "Retrieve all **ISWC** attendees"

Answers

Node_1	Node_2	Time_1	Time_2
Alice	ISWC	100	100
		101	101
		102	102

A TRPQ navigates in a graph, but also in time



Query

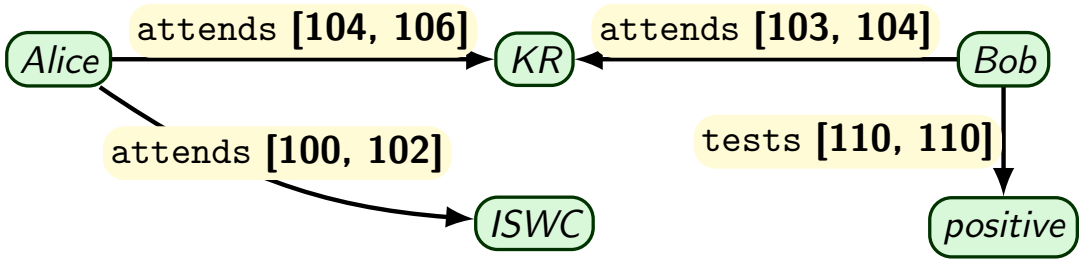
$(= ISWC)/\texttt{attends}^{-}$

- "Retrieve all ISWC attendees **(swapped)**"

Answers

Node_1	Node_2	Time_1	Time_2
ISWC	Alice	100	100
		101	101
		102	102

A TRPQ navigates in a graph, but also in time



Query

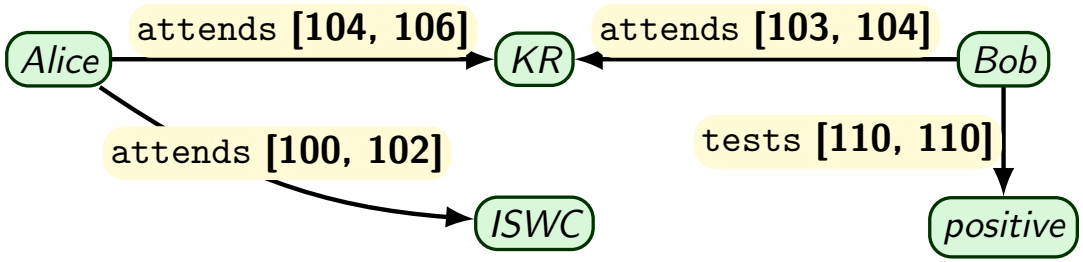
$(= ISWC)/\texttt{attends}^{-}/\mathbf{T}_{[3,4]}$

- "Retrieve all ISWC attendees (swapped) **and navigate in time by 3 to 4 days**"

Answers

Node_1	Node_2	Time_1	Time_2
ISWC	Alice	100	103
		100	104
		101	104
		101	105
		102	105
		102	106

A TRPQ navigates in a graph, but also in time



Query

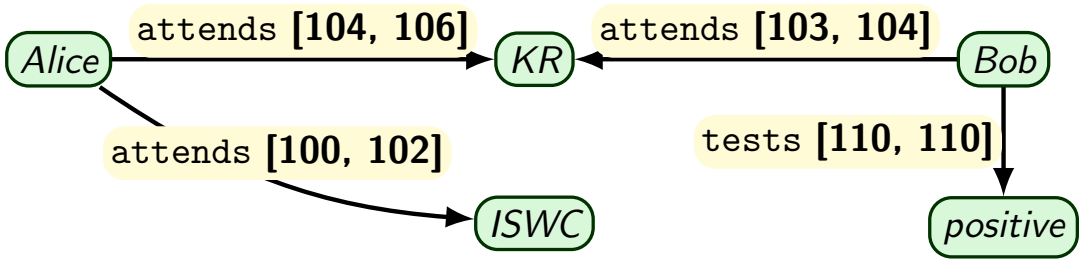
$(= ISWC)/\texttt{attends}^{-}/\mathbf{T}_{[3,4]}/\texttt{attends}$

- "Retrieve all conferences where the virus may have been transmitted by an ISWC attendee"

Answers

Node_1	Node_2	Time_1	Time_2
ISWC	KR	100	104
		101	104
		101	105
		102	105
		102	106

A TRPQ navigates in a graph, but also in time



Query

$$(= ISWC)/\texttt{attends}^{-}/\mathbf{T}_{[3,4]}/\texttt{attends}/\texttt{attends}^{-}/?\Big(\mathbf{T}_{[0,7]}/\texttt{test}/(= positive)\Big)$$

- "Retrieve all people who may have been infected by an ISWC attendee and tested positive at most a week after the infection"

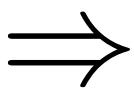
Answers

Node_1	Node_2	Time_1	Time_2
ISWC	Bob	100	104
		101	104

We use distances to represent temporal relations

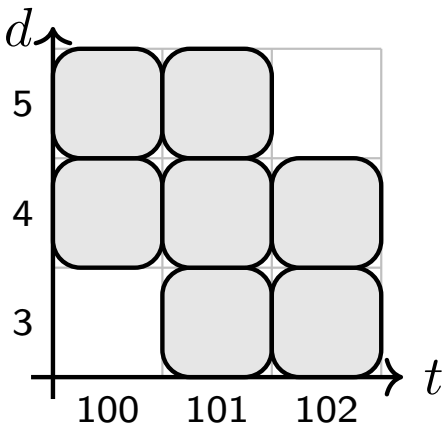
$$\texttt{attends}^{-}/(= \mathit{Alice})/\mathbf{T}_{[3,5]}/\texttt{attends}$$

$\mathbf{Node}_1$	$\mathbf{Node}_2$	$\mathbf{Time}_1$	$\mathbf{Time}_2$
		100	104
		100	105
		101	104
ISWC	KR	101	105
		101	106
		102	105
		102	106



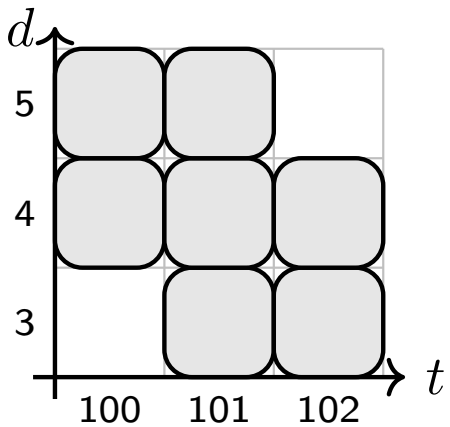
$\mathbf{Node}_1$	$\mathbf{Node}_2$	$\mathbf{Time}$	$\mathbf{Distance}$
		100	4
		100	5
		101	3
ISWC	KR	101	4
		101	5
		102	3
		102	4

We investigate four compact representations



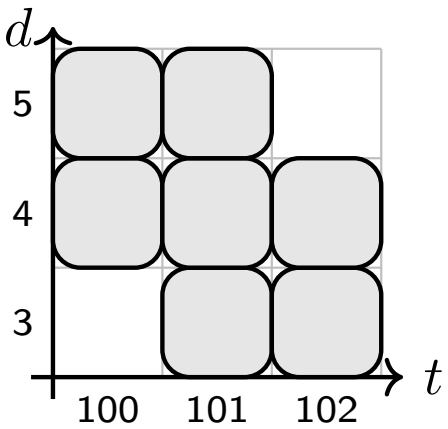
Time	Distance
100	4
100	5
101	3
101	4
101	5
102	3
102	4

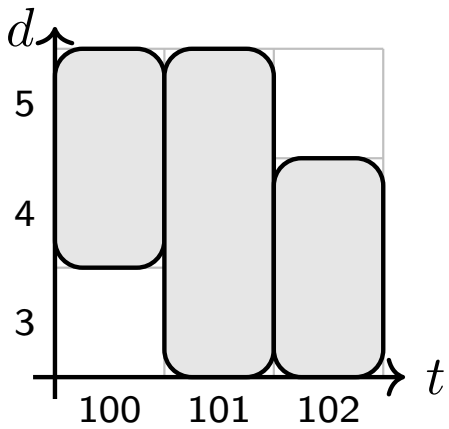
We investigate four compact representations



t	d
100	4
100	5
101	3
101	4
101	5
102	3
102	4

We investigate four compact representations



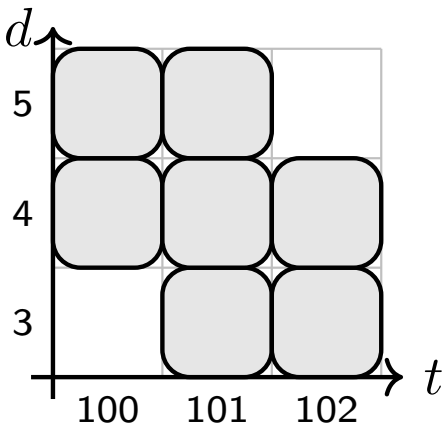


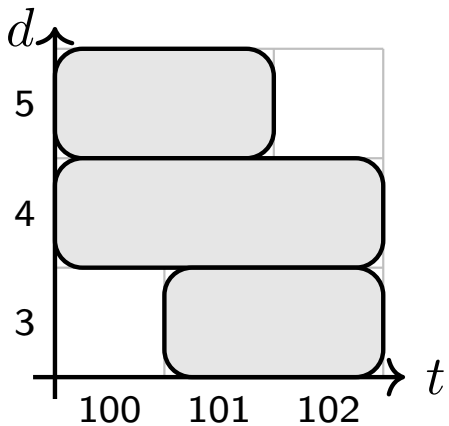
t	d
100	4
100	5
101	3
101	4
101	5
102	3
102	4

t	δ
100	[4,5]
101	[3,5]
102	[3,4]

- Group by **time**:
  - # answers affected by the size of **graph** intervals,
  - **cannot** handle dense time.

We investigate four compact representations



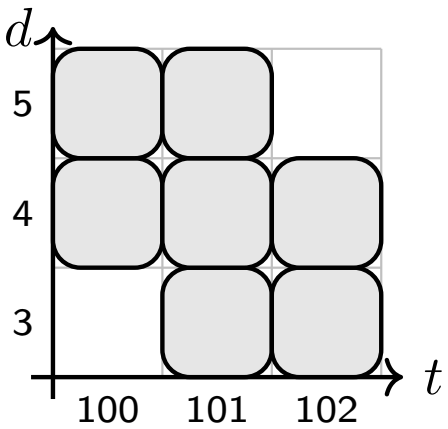


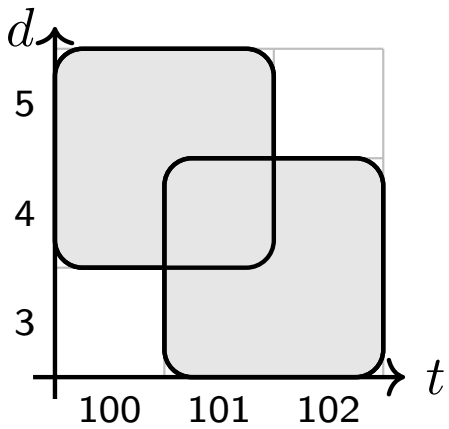
t	d
100	4
100	5
101	3
101	4
101	5
102	3
102	4

τ	d
[101,102]	3
[100,102]	4
[100,101]	5

- Group by **time**
- Group by **distance**:
  - # answers affected by the size of **query** intervals,
  - **cannot** handle dense time.

We investigate four compact representations



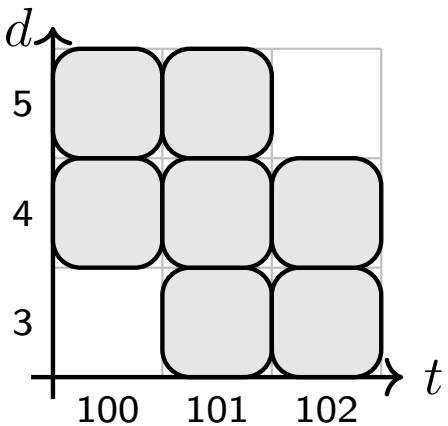


t	d
100	4
100	5
101	3
101	4
101	5
102	3
102	4

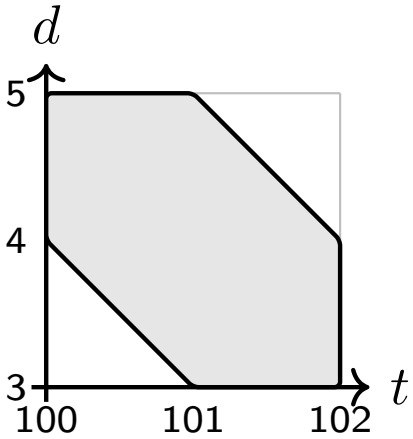
τ	δ
[100,101]	[4,5]
[101,102]	[3,4]

- Group by **time**
- Group by **distance**
- Group by **time and distance**:
  - **more compact**,

We investigate four compact representations



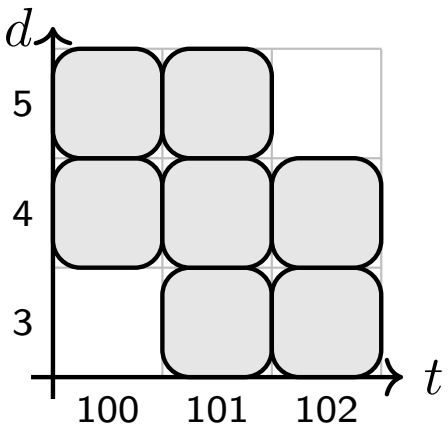
t	d
100	4
100	5
101	3
101	4
101	5
102	3
102	4

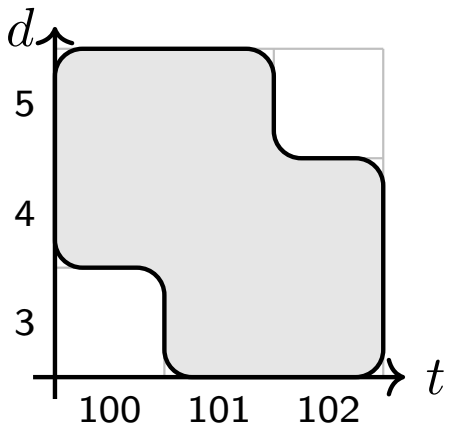


τ	δ
[100,101]	[4,5]
[101,102]	[3,4]

- Group by **time**
- Group by **distance**
- Group by **time and distance**:
  - **more compact**,
  - # answers affected by the size of **query** intervals,
  - **cannot** handle dense time.

We investigate four compact representations



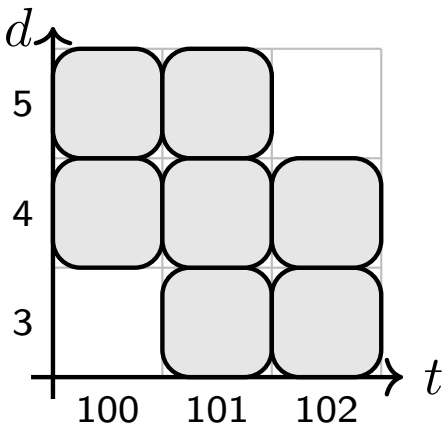


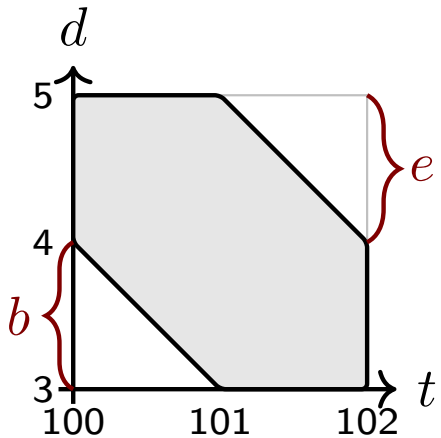
t	d
100	4
100	5
101	3
101	4
101	5
102	3
102	4

τ	δ	b	e
[100,102]	[3,5]	1	1

- Group by **time**
- Group by **distance**
- Group by **time and distance**
- Encode with **6 values**:
  - # answers **independent** of the size of (graph or query) intervals,
  - **can** handle dense time.

We investigate four compact representations





t	d
100	4
100	5
101	3
101	4
101	5
102	3
102	4

τ	δ	b	e
[100,102]	[3,5]	1	1

- Group by **time**
- Group by **distance**
- Group by **time and distance**
- Encode with **6 values**:
  - # answers **independent** of the size of (graph or query) intervals,
  - **can** handle dense time.

There is no free lunch

Format	Finite (dense time)	Unique	Size (closure-free q)		Minimization	Query answering
			time	distance		
no grouping	yes	yes	$\Omega(n)$	$\Omega(n)$	$O(n \log n)$	PSpace-c

There is no free lunch

Format	Finite (dense time)	Unique	Size (closure-free q)		Minimization	Query answering
			time	distance		
no grouping	yes	yes	$\Omega(n)$	$\Omega(n)$	$O(n \log n)$	PSpace-c
by time	no	yes	$\Omega(n)$	$O(1)$	$O(n \log n)$	PSpace-c
by distance	no	yes	$O(1)$	$\Omega(n)$	$O(n \log n)$	PSpace-c

- Grouping results alongside a single dimension:
  - the number of answers is still **affected** by the size of the input intervals in the other dimension,
  - this **rules out** dense time,
  - the most compact representation of the answers is **unique**,
  - computing it (out of a non-optimal one) is **tractable**.

There is no free lunch

Format	Finite (dense time)	Unique	Size (closure-free q)		Minimization	Query answering
			time	distance		
no grouping	yes	yes	$\Omega(n)$	$\Omega(n)$	$O(n \log n)$	PSpace-c
by time	no	yes	$\Omega(n)$	$O(1)$	$O(n \log n)$	PSpace-c
by distance	no	yes	$O(1)$	$\Omega(n)$	$O(n \log n)$	PSpace-c
by both	no	no	$O(1)$	$\Omega(n)$	NP-h / $O(n^{2.5})$	PSpace-c

- Grouping results alongside both dimensions:
  - the number of answers is **reduced**,
  - but still **affected** by the size of distance intervals,
  - this **rules out** dense time,
  - the most compact representation of the answers is **not unique**,
  - computing one is **intractable**,
  - unless one **disallows overlapping answers**.

There is no free lunch

Format	Finite (dense time)	Unique	Size (closure-free q)		Minimization	Query answering
			time	distance		
no grouping	yes	yes	$\Omega(n)$	$\Omega(n)$	$O(n \log n)$	PSpace-c
by time	no	yes	$\Omega(n)$	$O(1)$	$O(n \log n)$	PSpace-c
by distance	no	yes	$O(1)$	$\Omega(n)$	$O(n \log n)$	PSpace-c
by both	no	no	$O(1)$	$\Omega(n)$	NP-h / $O(n^{2.5})$	PSpace-c
6 values	yes	no	$O(1)$	$O(1)$	NP-h	PSpace-h

- Under the 6 values encoding:
  - the number of answers is **independent** of the size of the input intervals,
  - this **allows** dense time,
  - the most compact representation of the answers is **not unique**,
  - computing one is **intractable**.

There is no free lunch

Format	Finite (dense time)	Unique	Size (closure-free q)		Minimization	Query answering
			time	distance		
no grouping	yes	yes	$\Omega(n)$	$\Omega(n)$	$O(n \log n)$	PSpace-c
by time	no	yes	$\Omega(n)$	$O(1)$	$O(n \log n)$	PSpace-c
by distance	no	yes	$O(1)$	$\Omega(n)$	$O(n \log n)$	PSpace-c
by both	no	no	$O(1)$	$\Omega(n)$	NP-h / $O(n^{2.5})$	PSpace-c
6 values	yes	no	$O(1)$	$O(1)$	NP-h	PSpace-h

- **No increase** in (combined) complexity for query answering (decision problem) under the 3 first compact representations.

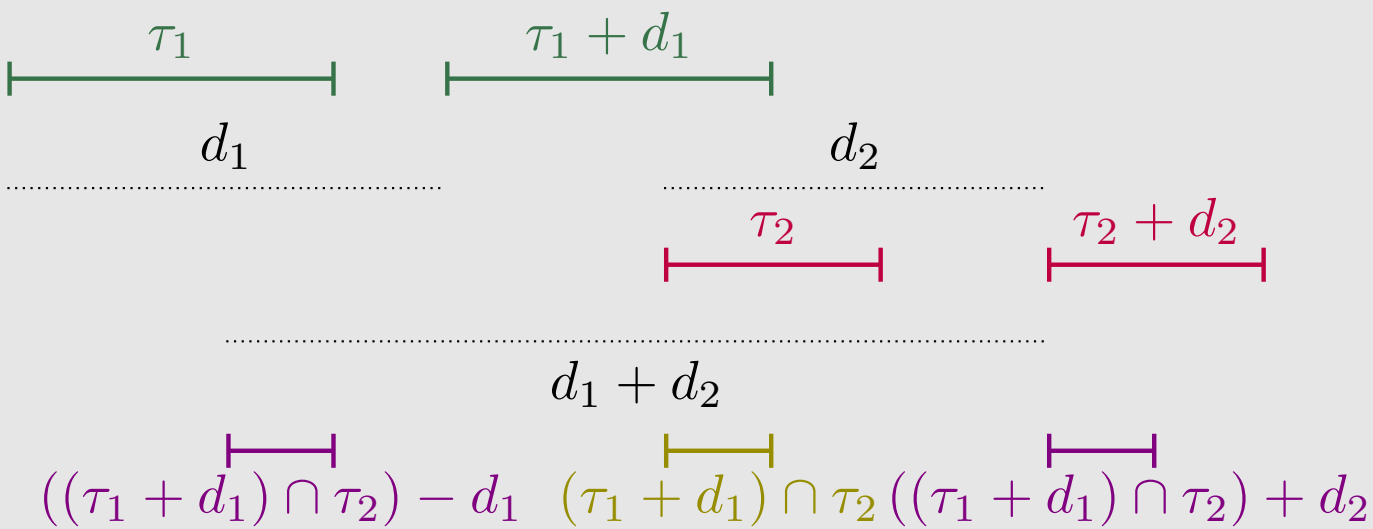
We specify a prototype implementation

- To prove our results, we provide an **inductive characterization** of each TRPQ operator, for each of the 4 representations.

We specify a prototype implementation

- To prove our results, we provide an **inductive characterization** of each TRPQ operator, for each of the 4 representations.

Example: relation composition (the "/" operator) for the 2nd representation



We specify a prototype implementation

- To prove our results, we provide an **inductive characterization** of each TRPQ operator, for each of the 4 representations.
- These definitions specify a possible **implementation**.

An efficient implementation requires further investigation

- Leverage appropriate **indexes** and/or data structures.

An efficient implementation requires further investigation

- Leverage appropriate **indexes** and/or data structures.
- Use a **backend**:
  - SQL engine,
  - graph database (e.g. Neo4J),
  - library for (extended) linear algebra (e.g. GraphBLAS),
  - etc.

An efficient implementation requires further investigation

- Leverage appropriate **indexes** and/or data structures.
- Use a **backend**:
  - SQL engine,
  - graph database (e.g. Neo4J),
  - library for (extended) linear algebra (e.g. GraphBLAS),
  - etc.

Proof of concept [Adnan et al., 2025]

We provide an implementation
for a fragment of TRPQs and the **first 2** representations,
with relation **minimization**,
over PostgreSQL
using techniques from temporal databases.

Continuations can be explored

- **Generalize** compact answers to:
  - **n-ary** non-temporal tuples,
  - **n-ary** temporal relations,
  - distance intervals **in the data** (rather than in the query only).

Continuations can be explored

- **Generalize** compact answers to:
  - **n-ary** non-temporal tuples,
  - **n-ary** temporal relations,
  - distance intervals **in the data** (rather than in the query only).
- **Reason** over temporal relations
  (rather than set of time points),
  behind query answering.

Thank you for your attention

References


Adnan, M., Calvanese, D., Corman, J., Dignös, A., Nutt, W., and Savković, O. (2025).
Computing compact answers to temporal path queries using sql.
In *RuleML+RR'25: Companion Proceedings of the 9th International Joint Conference on Rules and Reasoning*.


Arenas, M., Bahamondes, P., Aghasadeghi, A., and Stoyanovich, J. (2022).
Temporal regular path queries.
In *Proc. of the 38th IEEE Int. Conf. on Data Engineering (ICDE)*, pages 2412–2425.