Querying Temporal Anomalies in Healthcare Information Systems and Beyond

Christina Khnaisser [1] Hind Hamrouni [2]
David B. Blumenthal[3] Anton Dignös[2] Johann Gamper[2]

[1] Université de Sherbrooke, Sherbrooke, Canada
[2] Free University of Bozen-Bolzano, Bozen, Italy
[3] Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany

September 6th, 2022

Supported by Ministère de l'Économie et de l'Innovation – Québec and by the Autonomous Province of Bozen-Bolzano with research call "Research Südtirol/Alto Adige 2019" (project Enabling Industrial-Strength, Open-Source Temporal Query Processing – ISTeP)

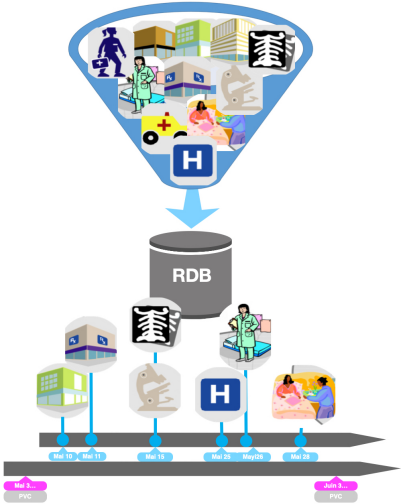
UDS Université de Sherbrooke

unibz

FAU FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG

Outline

Motivation & goal



- Temporal data integration is challenging task
- Anomalies could occur when feeding healthcare data system
- Detection of anomalies is crucial for consistency preserving and repairing

Contributions

- Defining the temporal anomalies:
  - temporal redundancy,
  - temporal contradiction,
  - temporal incompleteness.

Contributions

- Defining the temporal anomalies:
  - temporal redundancy,
  - temporal contradiction,
  - temporal incompleteness.
- Presenting two solutions for anomaly detection: **anomaly labelling** and **anomaly retrieval**.

Contributions

- Defining the temporal anomalies:
  - temporal redundancy,
  - temporal contradiction,
  - temporal incompleteness.
- Presenting two solutions for anomaly detection: **anomaly labelling** and **anomaly retrieval**.
- Implementing the solutions using temporal aggregation.

Contributions

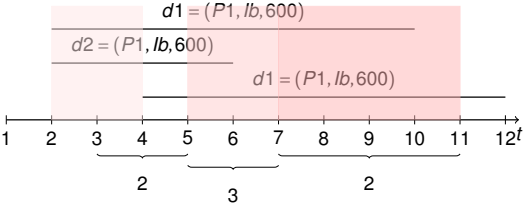
- Defining the temporal anomalies:
  - temporal redundancy,
  - temporal contradiction,
  - temporal incompleteness.
- Presenting two solutions for anomaly detection: **anomaly labelling** and **anomaly retrieval**.
- Implementing the solutions using temporal aggregation.
- Performing experiments on real-world medical data.

Temporal Redundancy

Same information stored at the same time point.

Prescription

	Pat	Name	Dose	T
d_1	P1	Ibuprofen	600mg	[2,10)
d_2	P1	Ibuprofen	600mg	[2,6)
d_3	P1	Ibuprofen	600mg	[4,12)
d_4	P2	Clopidogrel	75mg	[1,12)
d_5	P2	Clopidogrel	85mg	[10,14)
d_6	P2	Clopidogrel	95mg	[8,11)

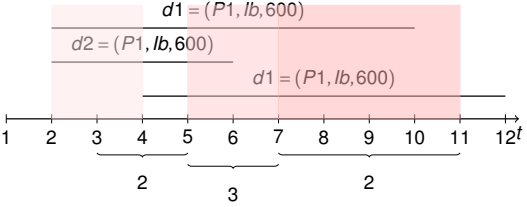


Temporal Redundancy

Same information stored at the same time point.

Prescription

	Pat	Name	Dose	T
d_1	P1	Ibuprofen	600mg	[2,10)
d_2	P1	Ibuprofen	600mg	[2,6)
d_3	P1	Ibuprofen	600mg	[4,12)
d_4	P2	Clopidogrel	75mg	[1,12)
d_5	P2	Clopidogrel	85mg	[10,14)
d_6	P2	Clopidogrel	95mg	[8,11)



Two solutions for temporal redundancy detection:

Pat	Name	Dose	T	Count
P1	Ibuprofen	600mg	[2,4)	2
P1	Ibuprofen	600mg	[4,6)	3
P1	Ibuprofen	600mg	[6,10)	2
P1	Ibuprofen	600mg	[10,12)	1
P2	Clopidogrel	75mg	[1,12)	1
P2	Clopidogrel	85mg	[10,14)	1
P2	Clopidogrel	95mg	[8,11)	1

(a) Redundancy labelling

Pat	Name	Dose	T	Count
P1	Ibuprofen	600mg	[2,4)	2
P1	Ibuprofen	600mg	[4,6)	3
P1	Ibuprofen	600mg	[6,10)	2

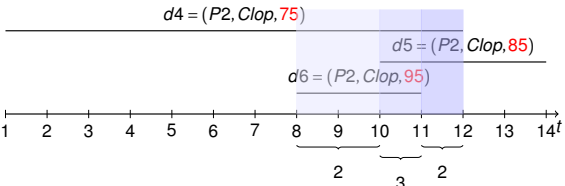
(b) Redundancy retrieval

Temporal Contradiction

Contradicting information stored at the same time point.

Prescription

	Pat	Name	Dose	T
d_1	P1	Ibuprofen	600mg	[2,10)
d_2	P1	Ibuprofen	600mg	[2,6)
d_3	P1	Ibuprofen	600mg	[4,12)
d_4	P2	Clopidogrel	75mg	[1,12)
d_5	P2	Clopidogrel	85mg	[10,14)
d_6	P2	Clopidogrel	95mg	[8,11)

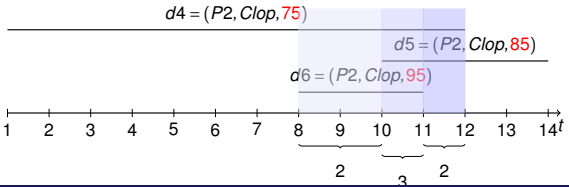


Temporal Contradiction

Contradicting information stored at the same time point.

Prescription

	Pat	Name	Dose	T
d_1	P1	Ibuprofen	600mg	[2,10)
d_2	P1	Ibuprofen	600mg	[2,6)
d_3	P1	Ibuprofen	600mg	[4,12)
d_4	P2	Clopidogrel	75mg	[1,12)
d_5	P2	Clopidogrel	85mg	[10,14)
d_6	P2	Clopidogrel	95mg	[8,11)



Two solutions for temporal contradiction detection:

Pat	Name	T	Contradiction
P1	Ibuprofen	[2,10)	600mg
P1	Ibuprofen	[2,6)	600mg
P1	Ibuprofen	[4,12)	600mg
P2	Clopidogrel	[1,8)	75mg
P2	Clopidogrel	[8,10)	75mg, 95mg
P2	Clopidogrel	[10,11)	75mg, 85mg, 95mg
P2	Clopidogrel	[11,12)	75mg, 85mg
P2	Clopidogrel	[12,14)	85mg

(a) Contradiction labelling

Pat	Name	T	Contradiction
P2	Clopidogrel	[8,10)	75mg, 95mg
P2	Clopidogrel	[10,11)	75mg, 85mg, 95mg
P2	Clopidogrel	[11,12)	75mg, 85mg

(b) Contradiction retrieval

Temporal Incompleteness

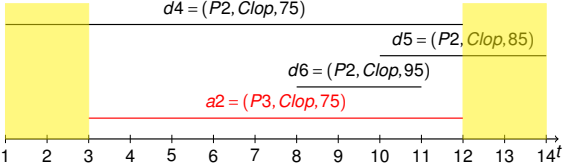
Missing information at the same point of time in the other relation.

Prescription

	Pat	Name	Dose	T
d_1	P1	Ibuprofen	600mg	[2,10)
d_2	P1	Ibuprofen	600mg	[2,6)
d_3	P1	Ibuprofen	600mg	[4,12)
d_4	P2	Clopidogrel	75mg	[1,12)
d_5	P2	Clopidogrel	85mg	[10,14)
d_6	P2	Clopidogrel	95mg	[8,11)

Administration

	Pat	Name	Dose	T
a_1	P1	Ibuprofen	600mg	[2,12)
a_2	P2	Clopidogrel	75mg	[3,12)



Temporal Incompleteness

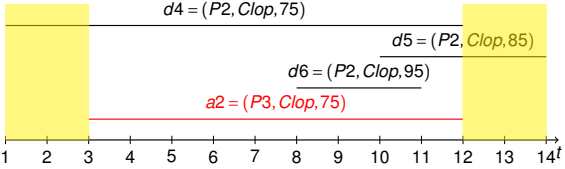
Missing information at the same point of time in the other relation.

Prescription

	Pat	Name	Dose	T
d_1	P1	Ibuprofen	600mg	[2,10)
d_2	P1	Ibuprofen	600mg	[2,6)
d_3	P1	Ibuprofen	600mg	[4,12)
d_4	P2	Clopidogrel	75mg	[1,12)
d_5	P2	Clopidogrel	85mg	[10,14)
d_6	P2	Clopidogrel	95mg	[8,11)

Administration

	Pat	Name	Dose	T
a_1	P1	Ibuprofen	600mg	[2,12)
a_2	P2	Clopidogrel	75mg	[3,12)



Two solutions for temporal incompleteness detection:

Pat	Name	T	Src
P1	Ibuprofen	[2,12)	presc, admin
P2	Clopidogrel	[1,3)	presc
P2	Clopidogrel	[3,12)	presc, admin
P2	Clopidogrel	[12,14)	presc

(a) Incompleteness labelling

Pat	Name	T	Src
P2	Clopidogrel	[1,3)	presc
P2	Clopidogrel	[12,14)	presc

(b) Incompleteness retrieval

SQL Implementations

- Providing three different implementations for the retrieval and labelling solutions of the three anomalies:
  - Unfold/Fold
  - Unfold/Fold Join Filtered
  - Window Function
- Presenting redundancy retrieval using the three implementations.

SQL Implementations

Unfold/Fold

(Lorentzos and Mitsopoulos 1997)

Pat	Name	Dose	T
P1	Ibuprofen	600mg	[1,4)
P1	Ibuprofen	600mg	[1,3)
P1	Ibuprofen	600mg	[2,5)

$\Longrightarrow$

UNFOLD

t
1
2
3
1
2
2
3
4

$\Longrightarrow$

Aggregation

t	count
{1, 1}	2
{2, 2, 2}	3
{3, 3}	2
{4}	1

$\Longrightarrow$

FOLD

T	count
[1,2)	2
[2,3)	3
[3,4)	2

SQL Implementations

Unfold/Fold

(Lorentzos and Mitsopoulos 1997)

Pat	Name	Dose	T
P1	Ibuprofen	600mg	[1,4)
P1	Ibuprofen	600mg	[1,3)
P1	Ibuprofen	600mg	[2,5)

$\Longrightarrow$

UNFOLD

t
1
2
3
1
2
2
3
4

$\Longrightarrow$

Aggregation

t	count
{1, 1}	2
{2, 2, 2}	3
{3, 3}	2
{4}	1

$\Longrightarrow$

FOLD

T	count
[1,2)	2
[2,3)	3
[3,4)	2

Unfold/Fold Join Filtered

- using a semi join to reduce the input for the unfold function.
- applied only for redundancy retrieval and contradiction retrieval.

SQL Implementations

Unfold/Fold

(Lorentzos and Mitsopoulos 1997)

Pat	Name	Dose	T
P1	Ibuprofen	600mg	[1,4)
P1	Ibuprofen	600mg	[1,3)
P1	Ibuprofen	600mg	[2,5)

$\Longrightarrow$ UNFOLD

t
1
2
3
1
2
2
3
4

$\Longrightarrow$ Aggregation

t	count
{1, 1}	2
{2, 2, 2}	3
{3, 3}	2
{4}	1

$\Longrightarrow$ FOLD

T	count
[1,2)	2
[2,3)	3
[3,4)	2

Unfold/Fold Join Filtered

- using a semi join to reduce the input for the unfold function.
- applied only for redundancy retrieval and contradiction retrieval.

Window Function

(Dignös et al. 2019)

Pat	Name	Dose	T
P1	Ibuprofen	600mg	[1,4)
P1	Ibuprofen	600mg	[1,3)
P1	Ibuprofen	600mg	[2,5)

$\Longrightarrow$

t	start	end
1	1	0
1	1	0
2	1	0
3	0	1
4	0	1
5	0	1

$\Longrightarrow$

t	count
1	1
1	2
2	3
3	2
4	1
5	0

$\Longrightarrow$

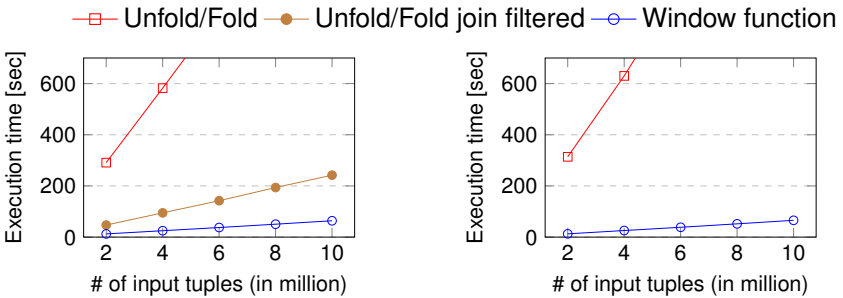
T	count
[1,2)	2
[2,3)	3
[3,4)	2

Experimental evaluation

- Dataset: Real-world MIMIC-IV (Medical Information Mart for Intensive Care) dataset (Johnson et al. 2020)
  - Prescriptions: 13M
  - Pharmacy: 10M
- Database Server: PostgreSQL version 14
- Measure: Comparison of runtime for anomaly retrieval and labeling with the three implementations

Experimental evaluation

Experiment 1: Runtime for temporal redundancy on the Prescription table.

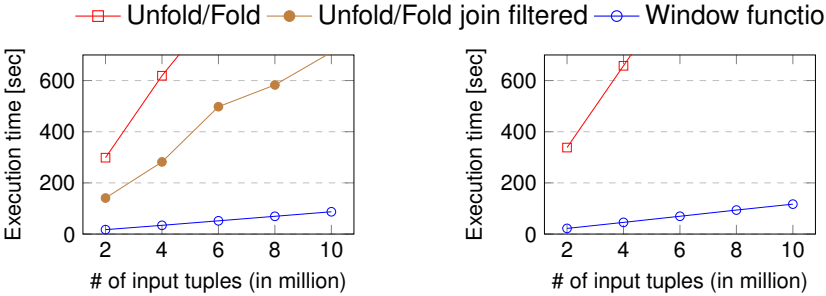


(a) Retrieval (b) Labelling

- Unfold/fold is very slow because unfold operation.
- Join filtering yields a substantial improvement in the performance.
- Window functions is the most efficient for retrieval and labelling.

Experimental evaluation

Experiment 2: Runtime for temporal contradiction on the Prescription table.

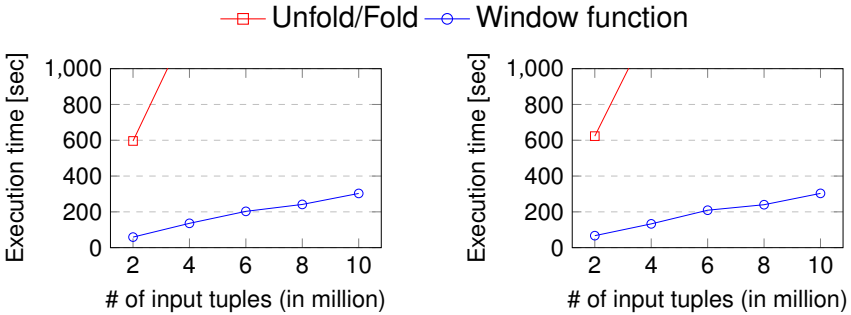


(a) Retrieval (b) Labeling

- The join filtering implementation is less effective compared to redundancy.
- Window functions is the most efficient.

Experimental evaluation

Experiment 3: Runtime for temporal incompleteness retrieval/labeling.



(a) Retrieval (b) Labelling

- Window functions is the most efficient.

Summary and Future work

Summary

- Defining three temporal anomalies: temporal redundancy, temporal contradiction and temporal incompleteness.
- Presenting temporal queries to detect inconsistencies.
- Providing efficient implementations.

Summary and Future work

Summary

- Defining three temporal anomalies: temporal redundancy, temporal contradiction and temporal incompleteness.
- Presenting temporal queries to detect inconsistencies.
- Providing efficient implementations.

Future work

- Extending the temporal anomalies operations to cover bitemporal models, cross-relational temporal contradictions, and semantic temporal conflicts.
- Developing a federated version for privacy-preserving anomaly mining.
- Defining a temporal repair system using healthcare expert preferences.

Thank you!