Towards Ideal Semantics for Analyzing Stream Reasoning

Harald Beck Minh Dao-Tran Thomas Eiter Michael Fink

International Workshop on Reactive Concepts in Knowledge Representation 2014

August 19, 2014

FAKULTÄT
FÜR !NFORMATIK

Faculty of Informatics

What & Why

“Towards Ideal Semantics for Analyzing Stream Reasoning”

- Stream Reasoning

What & Why

“Towards Ideal Semantics for Analyzing Stream Reasoning”

- Stream Reasoning: Logical reasoning on streaming data

What & Why

“Towards Ideal Semantics for Analyzing Stream Reasoning”

- Stream Reasoning: Logical reasoning on streaming data
  - Streams = **tuples** (atoms) with **timestamps**
  - Essential aspect: **window** functions

What & Why

“Towards Ideal Semantics for Analyzing Stream Reasoning”

- Stream Reasoning: Logical reasoning on streaming data
  - Streams = **tuples** (atoms) with **timestamps**
  - Essential aspect: **window** functions
- Semantics

What & Why

“Towards Ideal Semantics for Analyzing Stream Reasoning”

- Stream Reasoning: Logical reasoning on streaming data
  - Streams = **tuples** (atoms) with **timestamps**
  - Essential aspect: **window** functions
- Semantics: Lack of theory

What & Why

“Towards Ideal Semantics for Analyzing Stream Reasoning”

- Stream Reasoning: Logical reasoning on streaming data
  - Streams = **tuples** (atoms) with **timestamps**
  - Essential aspect: **window** functions
- Semantics: Lack of theory
- Analysis

What & Why

"Towards Ideal Semantics for Analyzing Stream Reasoning"

- Stream Reasoning: Logical reasoning on streaming data
  - Streams = **tuples** (atoms) with **timestamps**
  - Essential aspect: **window** functions
- Semantics: Lack of theory
- Analysis: Hard to predict, hard to compare

What & Why

“Towards Ideal Semantics for Analyzing Stream Reasoning”

- Stream Reasoning: Logical reasoning on streaming data
  - Streams = **tuples** (atoms) with **timestamps**
  - Essential aspect: **window** functions
- Semantics: Lack of theory
- Analysis: Hard to predict, hard to compare
- Ideal

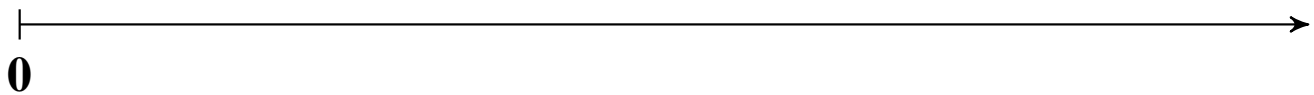
What & Why

"Towards Ideal Semantics for Analyzing Stream Reasoning"

- Stream Reasoning: Logical reasoning on streaming data
  - Streams = **tuples** (atoms) with **timestamps**
  - Essential aspect: **window** functions
- Semantics: Lack of theory
- Analysis: Hard to predict, hard to compare
- Ideal
  - Idealization: Abstract from practical (operational) issues
  - Generalization: Uniform representation

Example: Trams and buses

Arrival times at different stations p_i



Example: Trams and buses

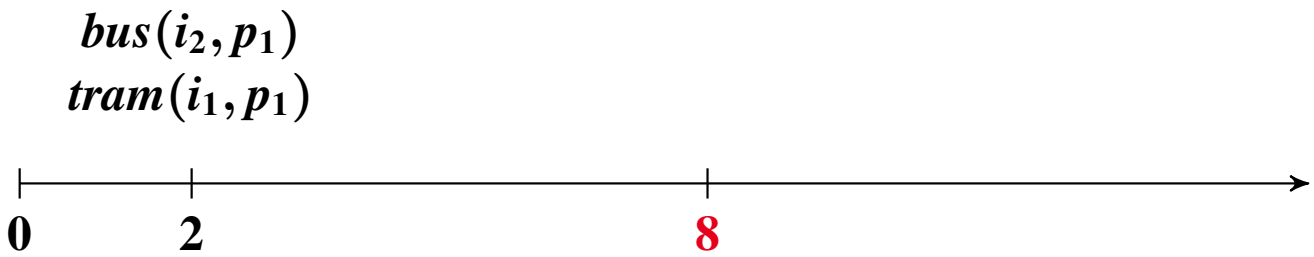
Arrival times at different stations p_i

Example: Trams and buses

Arrival times at different stations p_i

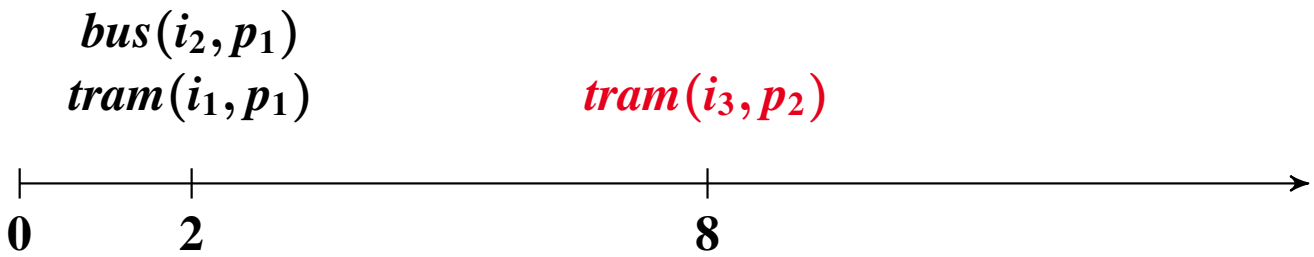
Example: Trams and buses

Arrival times at different stations p_i



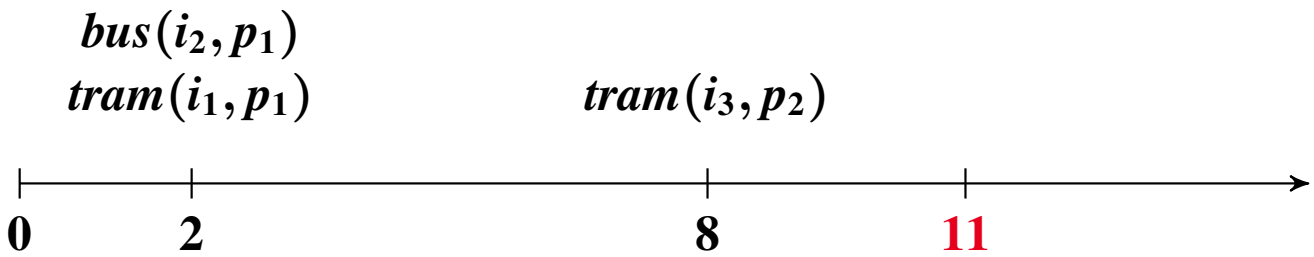
Example: Trams and buses

Arrival times at different stations p_i



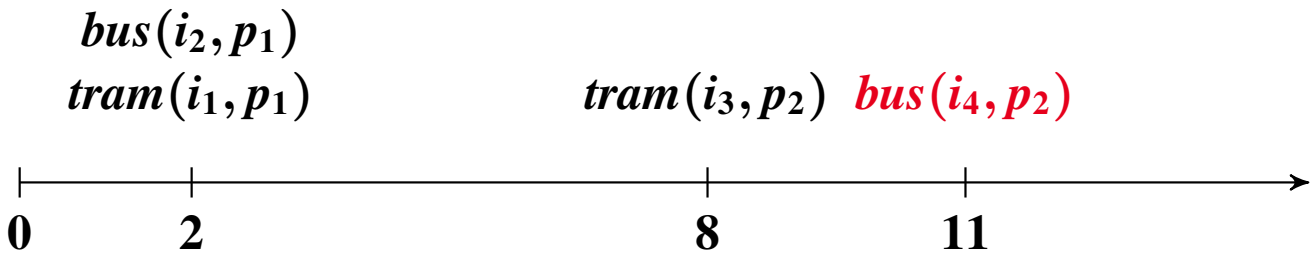
Example: Trams and buses

Arrival times at different stations p_i



Example: Trams and buses

Arrival times at different stations p_i



Example: Trams and buses

Arrival times at different stations p_i

$$bus(i_2, p_1)$$
$$tram(i_1, p_1) \qquad tram(i_3, p_2) \;\; bus(i_4, p_2)$$

- Normal DB: Query for

Example: Trams and buses

Arrival times at different stations p_i

$$bus(i_2, p_1)$$
$$tram(i_1, p_1) \qquad tram(i_3, p_2) \quad bus(i_4, p_2)$$

- Normal DB: Query for trams and buses arriving at same station P

Example: Trams and buses

Arrival times at different stations p_i

$bus(i_2, p_1)$
$tram(i_1, p_1)$

$tram(i_3, p_2)$ $bus(i_4, p_2)$

- Normal DB: Query for trams and buses arriving at same station P
  Answer: i_1, i_2, p_1

Example: Trams and buses

Arrival times at different stations p_i

$bus(i_2, p_1)$
$tram(i_1, p_1)$

$tram(i_3, p_2)$ $bus(i_4, p_2)$

- Normal DB: Query for trams and buses arriving at same station P
  Answer: i_1, i_2, p_1 and i_3, i_4, p_2

Example: Trams and buses

Arrival times at different stations p_i

$$bus(i_2, p_1) \quad tram(i_1, p_1) \qquad tram(i_3, p_2) \quad bus(i_4, p_2)$$

- Normal DB: Query for trams and buses arriving at same station P
  Answer: i_1, i_2, p_1 and i_3, i_4, p_2
- SQL

```
SELECT * FROM tram, bus
WHERE tram.P = bus.P
```

Example: Trams and buses

Arrival times at different stations p_i

$bus(i_2, p_1)$
$tram(i_1, p_1)$ $tram(i_3, p_2)$ $bus(i_4, p_2)$

- Normal DB: Query for trams and buses arriving at same station P
  Answer: i_1, i_2, p_1 and i_3, i_4, p_2
- SQL

```
SELECT * FROM tram, bus
WHERE tram.P = bus.P
```

Example: Trams and buses

Arrival times at different stations p_i

$bus(i_2, p_1)$
$tram(i_1, p_1)$ $tram(i_3, p_2)$ $bus(i_4, p_2)$

- Normal DB: Query for trams and buses arriving at same station P
  Answer: i_1, i_2, p_1 and i_3, i_4, p_2
- SQL

```
SELECT * FROM tram, bus
WHERE tram.P = bus.P
```

Example: Trams and buses

Arrival times at different stations p_i

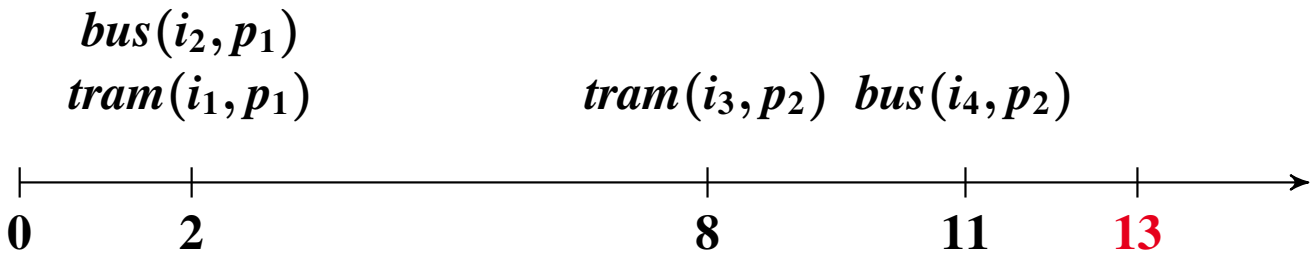
$$bus(i_2, p_1)$$
$$tram(i_1, p_1) \qquad tram(i_3, p_2)\ \ bus(i_4, p_2)$$

- Normal DB: Query for trams and buses arriving at same station P
  Answer: i_1, i_2, p_1 and i_3, i_4, p_2
- SQL

```
SELECT * FROM tram, bus
WHERE tram.P = bus.P
```

Example: Trams and buses

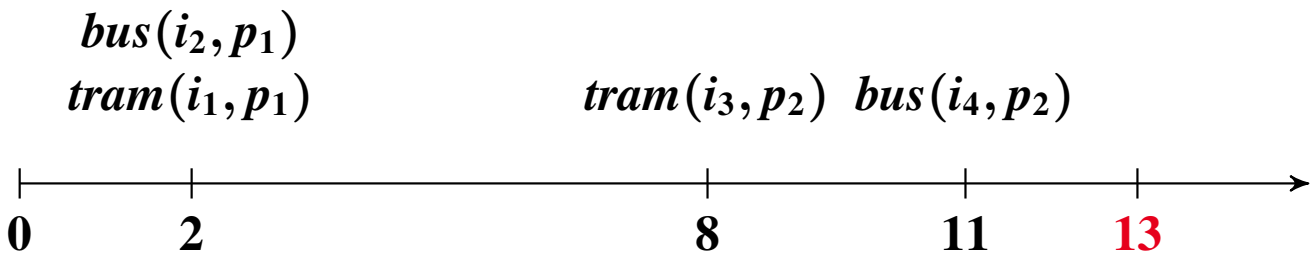
Arrival times at different stations p_i



- Stream setting, at time 13: Query for

Example: Trams and buses

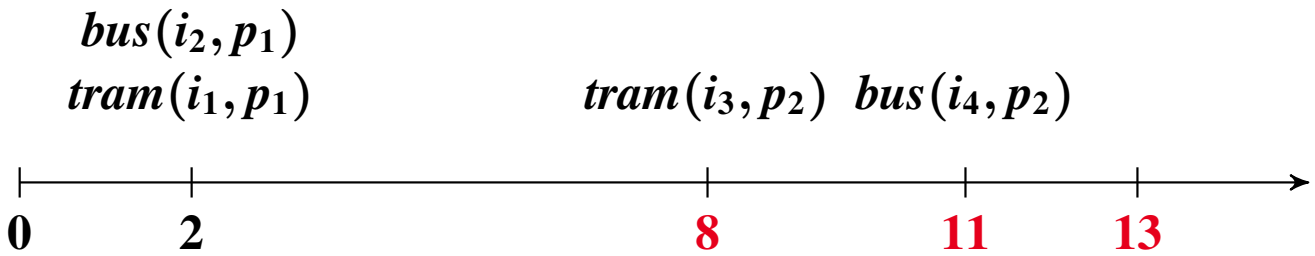
Arrival times at different stations p_i



- Stream setting, at time 13: Query for
- Trams and buses arriving at same station P

Example: Trams and buses

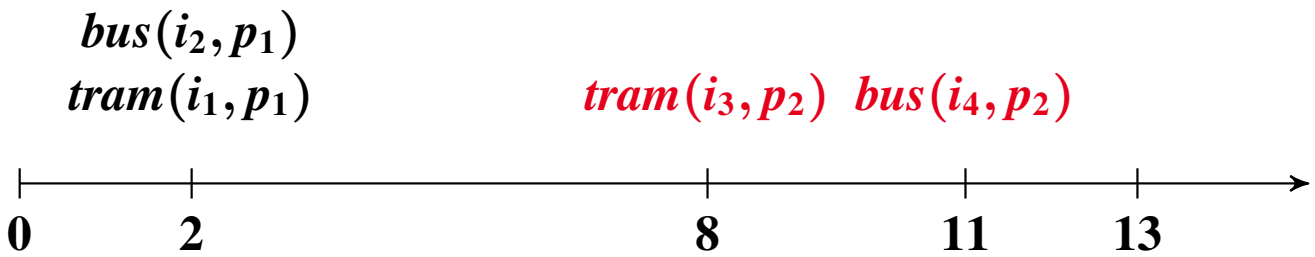
Arrival times at different stations p_i



- Stream setting, at time 13: Query for
- Trams and buses arriving at same station P within the last 5 min

Example: Trams and buses

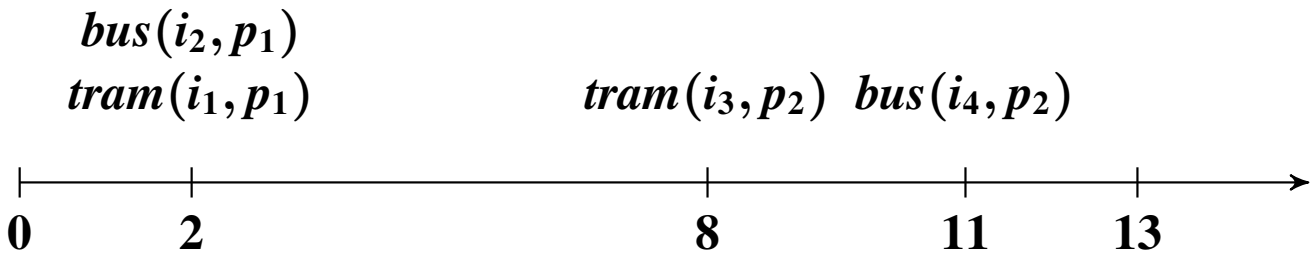
Arrival times at different stations p_i



- Stream setting, at time 13: Query for
- Trams and buses arriving at same station P within the last 5 min
  Answer: i_3, i_4, p_2

Example: Trams and buses

Arrival times at different stations p_i

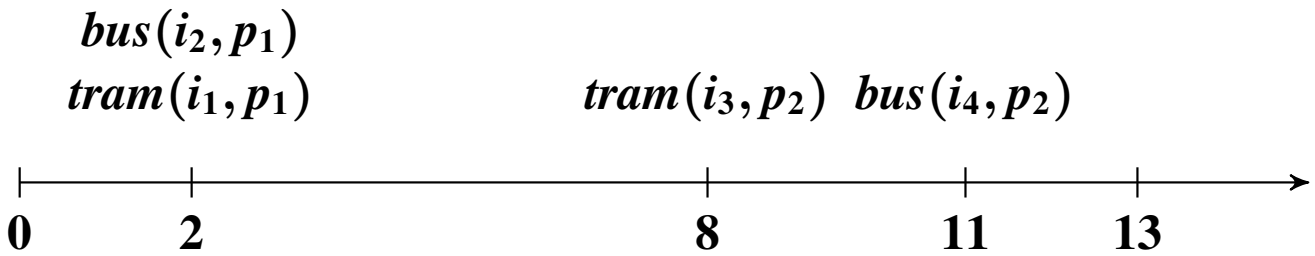


- Stream setting, at time 13: Query for
- Trams and buses arriving at same station P within the last 5 min
  Answer: i_3, i_4, p_2
- CQL

```
SELECT * FROM tram [RANGE 5], bus [RANGE 5]
WHERE tram.P = bus.P
```

Example: Trams and buses

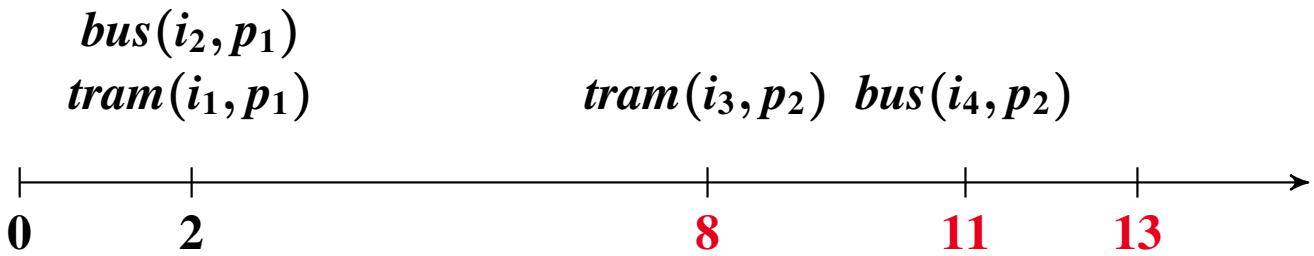
Arrival times at different stations p_i



- Trams and buses arriving at same station P within the last 5 min *at the same time*

Example: Trams and buses

Arrival times at different stations p_i

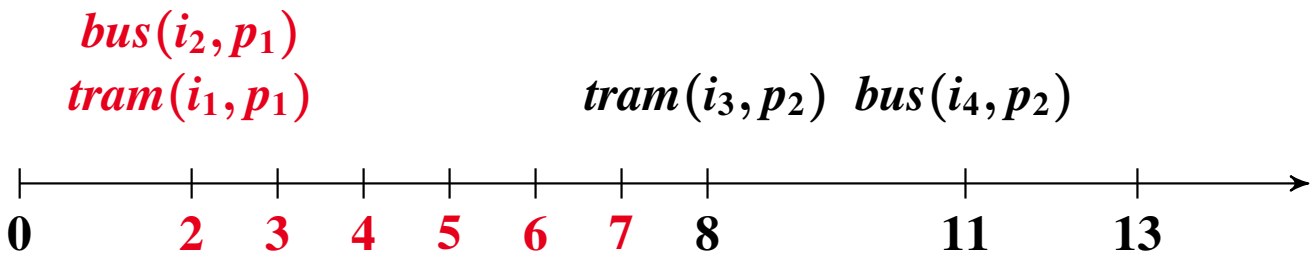


- Trams and buses arriving at same station P within the last 5 min *at the same time*

  Answer: –

Example: Trams and buses

Arrival times at different stations p_i

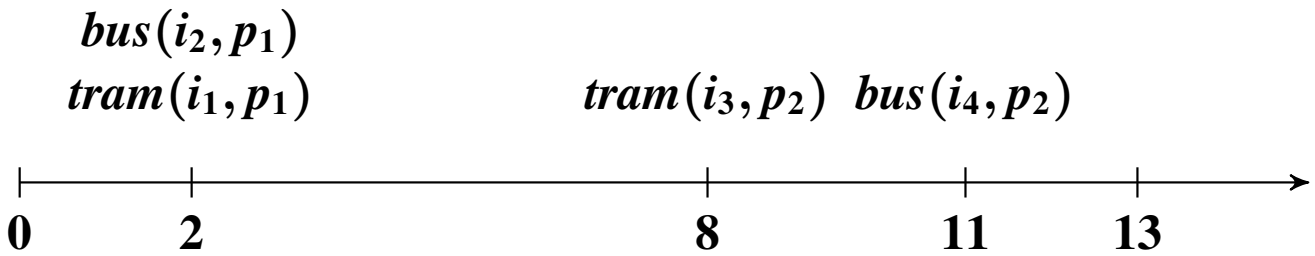


- Trams and buses arriving at same station P within the last 5 min *at the same time*

  Answer: i_1, i_2, p_1 for query times $2, \ldots, 7$

Example: Trams and buses

Arrival times at different stations p_i



- Trams and buses arriving at same station P within the last 5 min *at the same time*

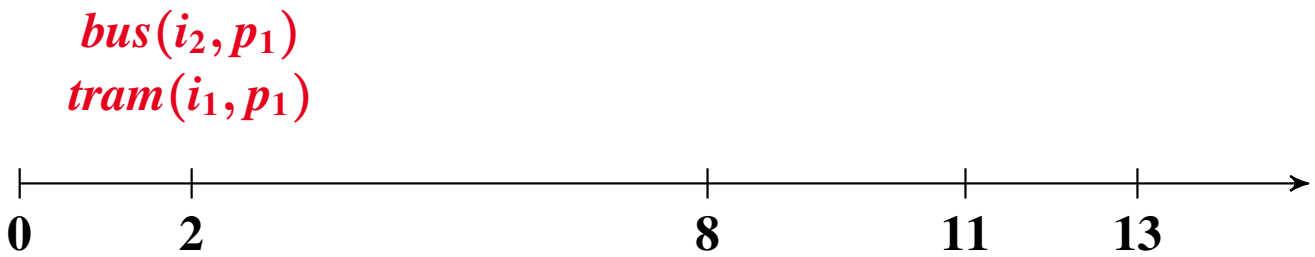
  Answer: i_1, i_2, p_1 for query times $2, \ldots, 7$

- CQL: Not expressible in single query (Snapshot semantics)

```
SELECT * AS tram_bus FROM tram [NOW], bus [NOW]
WHERE tram.P = bus.P
```

Example: Trams and buses

Arrival times at different stations p_i



- Trams and buses arriving at same station P within the last 5 min *at the same time*

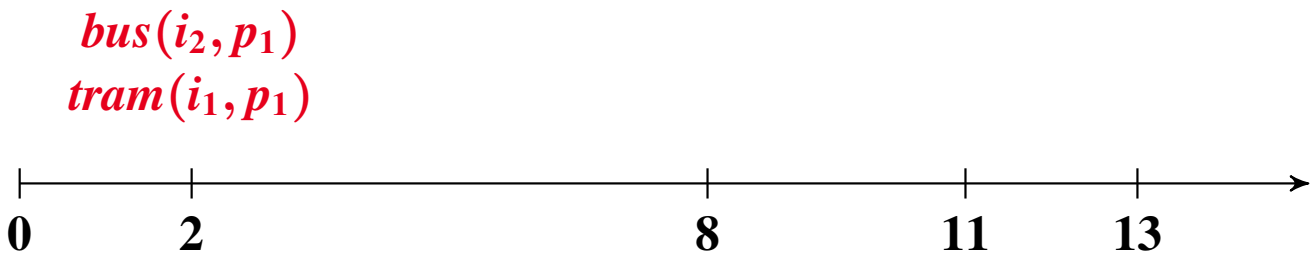
  Answer: i_1, i_2, p_1 for query times $2, \ldots, 7$

- CQL: Not expressible in single query (Snapshot semantics)

```
SELECT * AS tram_bus FROM tram [NOW], bus [NOW]
WHERE tram.P = bus.P
```

Example: Trams and buses

Arrival times at different stations p_i



- Trams and buses arriving at same station P within the last 5 min *at the same time*

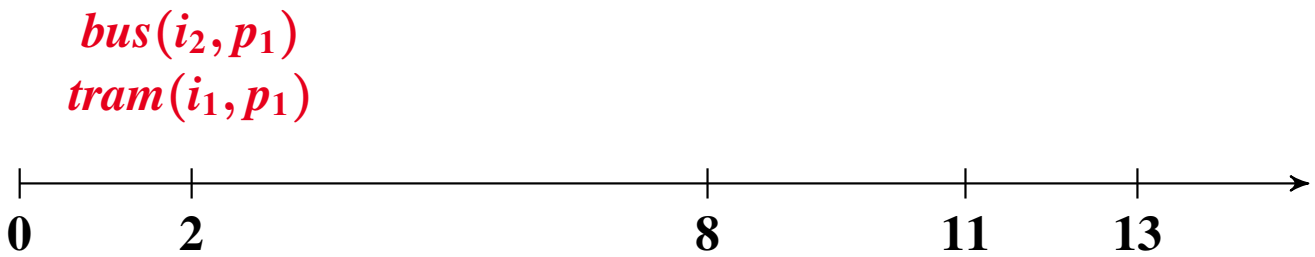
  Answer: i_1, i_2, p_1 for query times $2, \ldots, 7$

- CQL: Not expressible in single query (Snapshot semantics)

```
SELECT * AS tram_bus FROM tram [NOW], bus [NOW]
WHERE tram.P = bus.P

SELECT * FROM tram_bus [RANGE 5]
```

Example: Trams and buses

Arrival times at different stations p_i



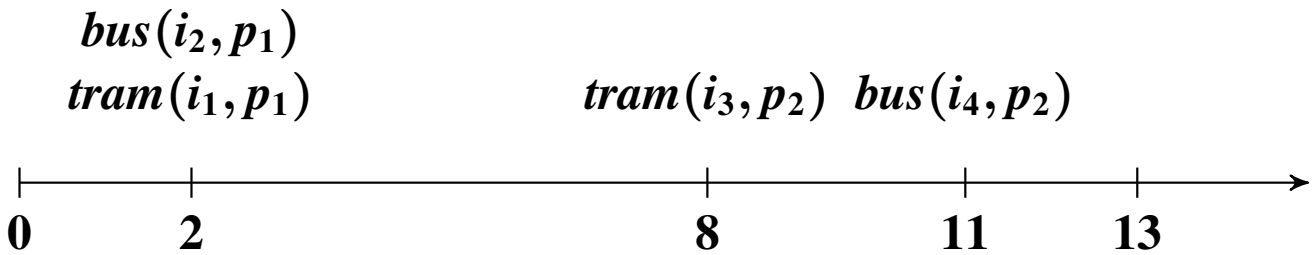
- Trams and buses arriving at same station P within the last 5 min *at the same time*

  Answer: i_1, i_2, p_1 for query times $2, \ldots, 7$

- CQL: Not expressible in single query (Snapshot semantics)

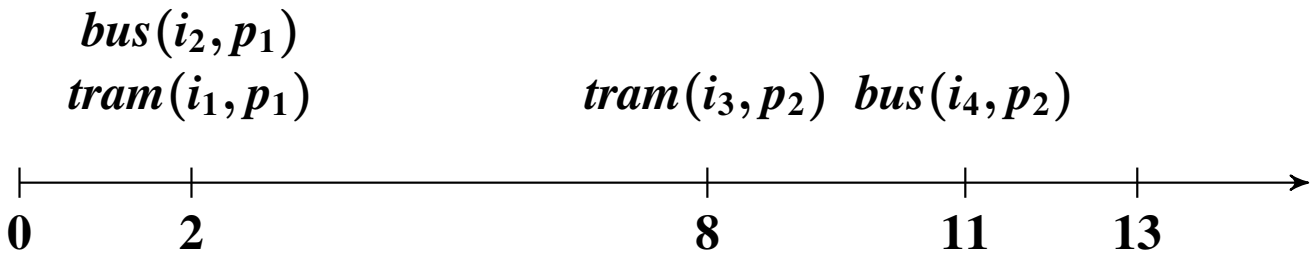
```
SELECT * AS tram_bus FROM tram [NOW], bus [NOW]
WHERE tram.P = bus.P

SELECT * FROM tram_bus [RANGE 5]
```

Window Types



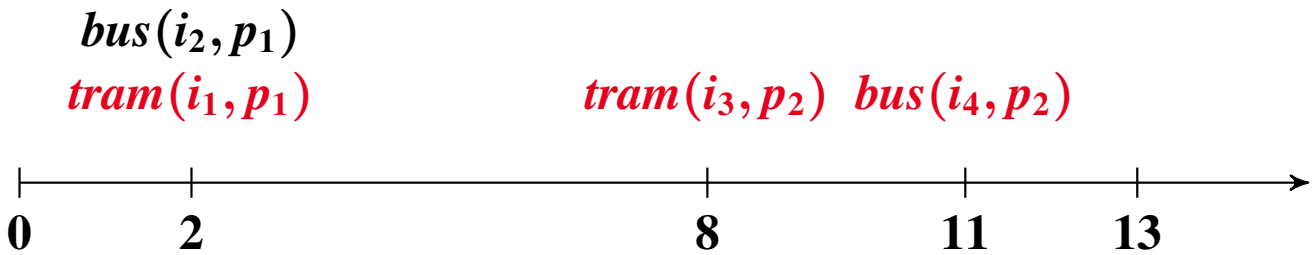
- Time-based

Window Types



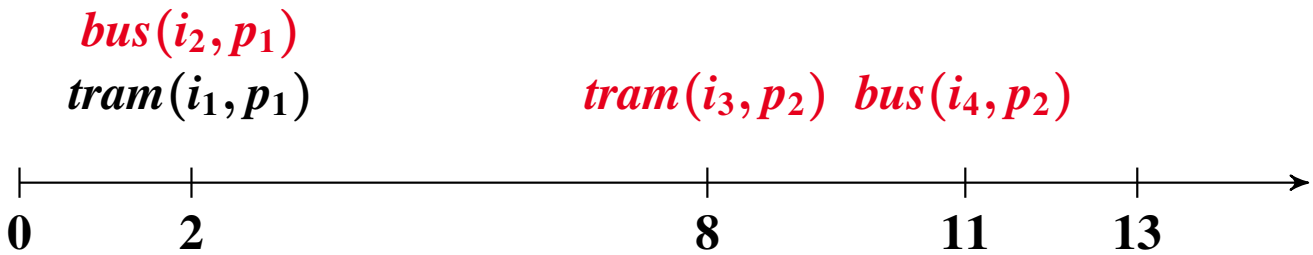
- Time-based
- Tuple-based

Window Types



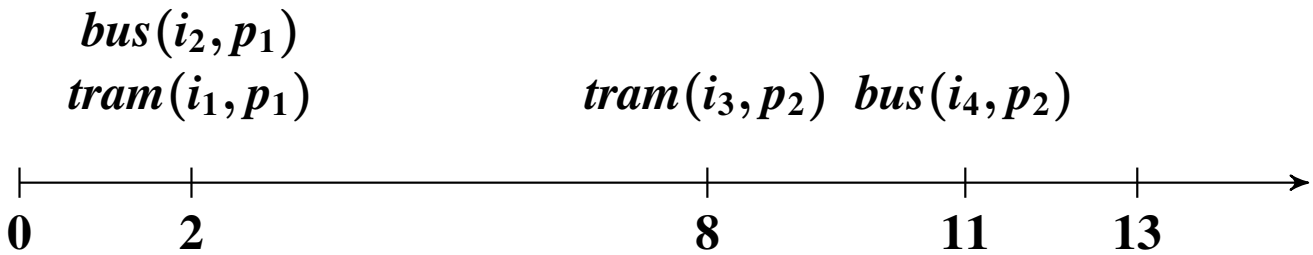
- Time-based
- Tuple-based
  - Not necessarily unique. E.g.: Last 3 tuples

Window Types



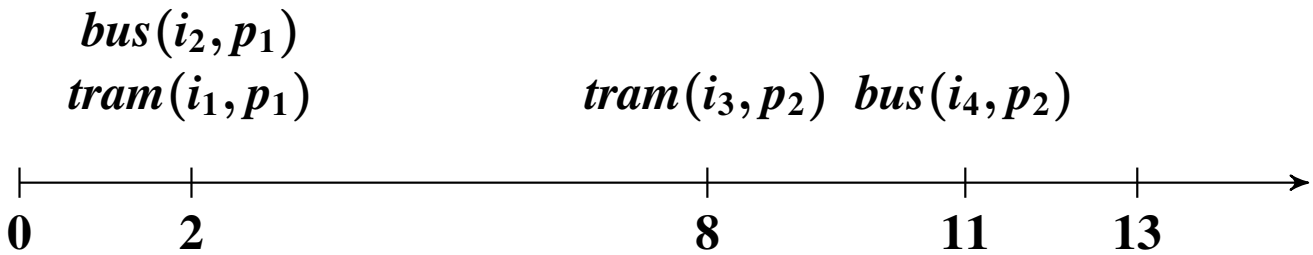
- Time-based
- Tuple-based
  - Not necessarily unique. E.g.: Last 3 tuples

Window Types



- Time-based
- Tuple-based
  - Not necessarily unique. E.g.: Last 3 tuples
- Partition-based

Window Types



- Time-based
- Tuple-based
  - Not necessarily unique. E.g.: Last 3 tuples
- Partition-based
  - Apply tuple-based window on substreams

Ideas for Windows

- Example: “In the last hour, did a bus always arrive within 5 min?”

Ideas for Windows

- Example: “In the last hour, did a bus always arrive within 5 min?”

Ideas for Windows

- Example: “In the last hour, did a bus always arrive within 5 min?”
- Allow for nesting: windows within windows
  - As formal counterpart to repeated runs of continuous queries

Ideas for Windows

- Example: “In the last hour, did a bus always arrive within 5 min?”
- Allow for nesting: windows within windows
  - As formal counterpart to repeated runs of continuous queries
- Allow for looking into the future

Ideas for Windows

- Example: “In the last hour, did a bus always arrive within 5 min?”
- Allow for nesting: windows within windows
  - As formal counterpart to repeated runs of continuous queries
- Allow for looking into the future
- View window operators as first class citizens
  - Do not separate window application (first) from logic (then)

Ideas for Windows

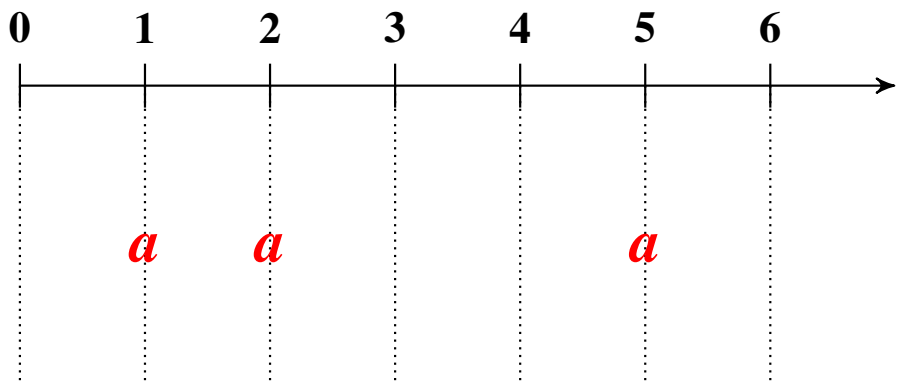
- Example: “In the last hour, did a bus always arrive within 5 min?”
- Allow for nesting: windows within windows
  - As formal counterpart to repeated runs of continuous queries
- Allow for looking into the future
- View window operators as first class citizens
  - Do not separate window application (first) from logic (then)
- Leave open specific underlying window functions

Ideas for Windows

- Example: “In the last hour, did a bus always arrive within 5 min?”
- Allow for nesting: windows within windows
  - As formal counterpart to repeated runs of continuous queries
- Allow for looking into the future
- View window operators as first class citizens
  - Do not separate window application (first) from logic (then)
- Leave open specific underlying window functions
  - $w(S, t) \mapsto S'$
  - Stream S, time point $t \in \mathbb{N}$, new stream S'

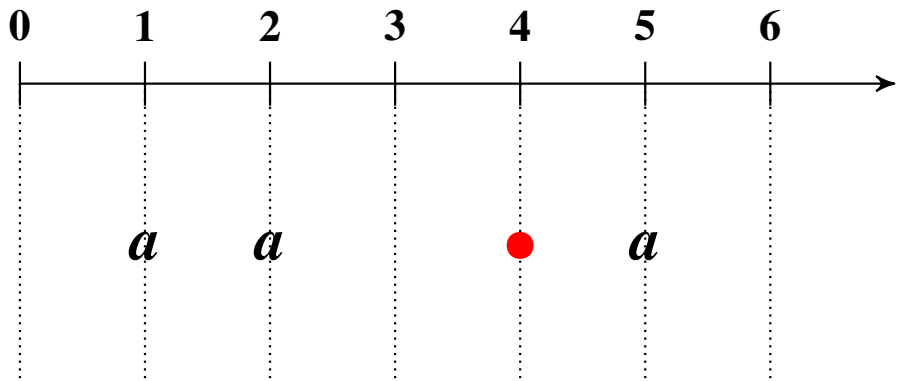
Ideas for Time Reference

- Atoms a appearing in the stream at time points $1, 2, 5$



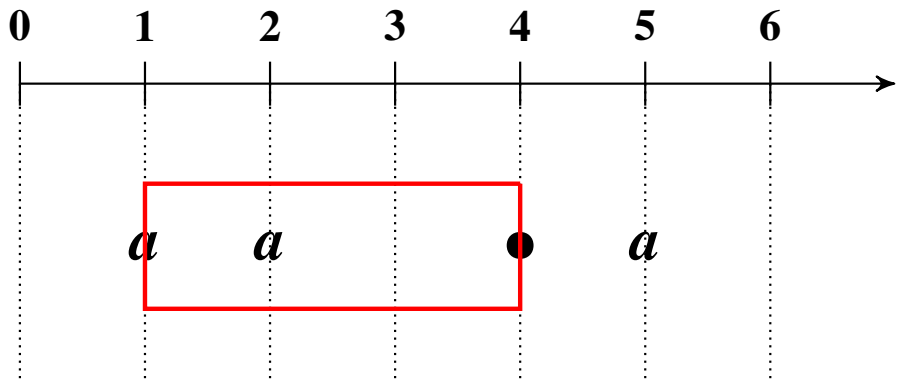
Ideas for Time Reference

- Atoms a appearing in the stream at time points $1, 2, 5$
- Query time $t = 4$.



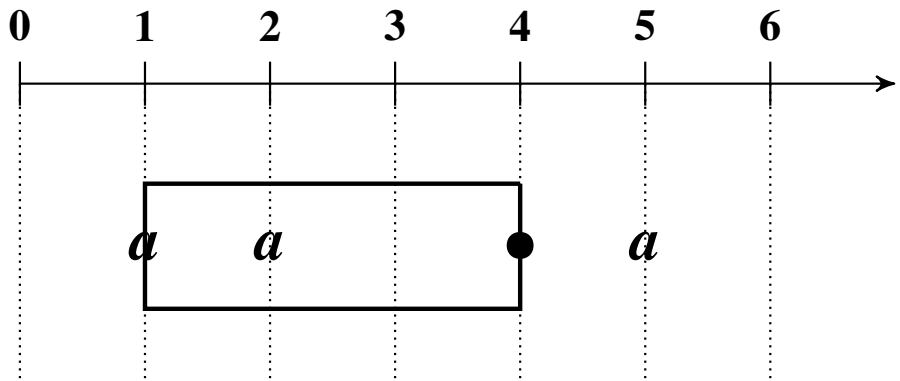
Ideas for Time Reference

- Atoms a appearing in the stream at time points $1, 2, 5$
- Query time $t = 4$. Window on interval $[1, 4]$



Ideas for Time Reference

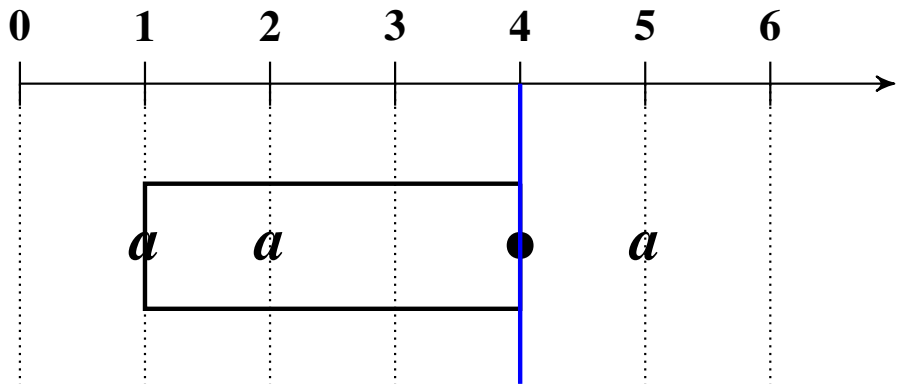
- Atoms a appearing in the stream at time points $1, 2, 5$
- Query time $t = 4$. Window on interval $[1, 4]$



- Example queries: In this window, does a hold. . .

Ideas for Time Reference

- Atoms a appearing in the stream at time points $1, 2, 5$
- Query time $t = 4$. Window on interval $[1, 4]$

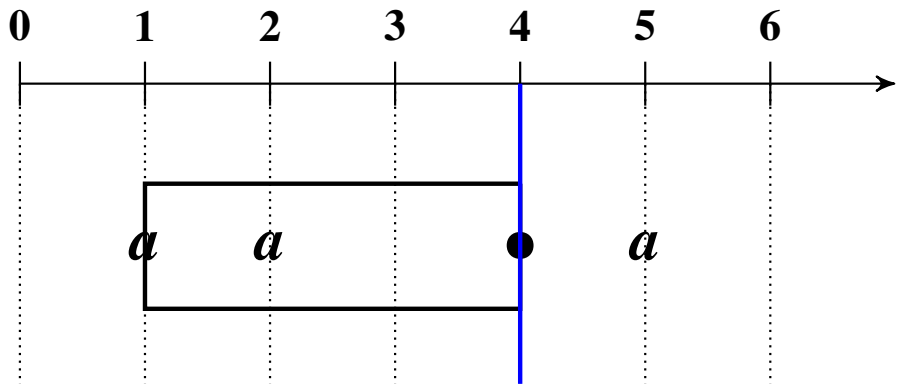


- Example queries: In this window, does a hold. . .

  . . . now, i.e., exactly at t?

Ideas for Time Reference

- Atoms a appearing in the stream at time points $1, 2, 5$
- Query time $t = 4$. Window on interval $[1, 4]$

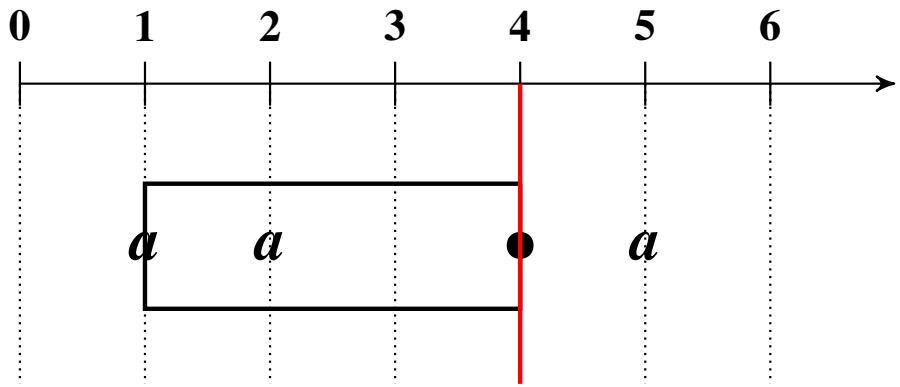


- Example queries: In this window, does a hold. . .

  . . . now, i.e., exactly at t? a

Ideas for Time Reference

- Atoms a appearing in the stream at time points $1, 2, 5$
- Query time $t = 4$. Window on interval $[1, 4]$

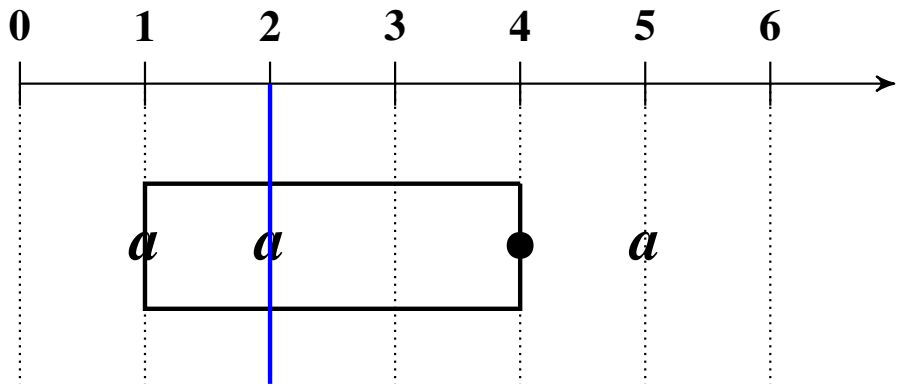


- Example queries: In this window, does a hold. . .

  . . . now, i.e., exactly at t? a no

Ideas for Time Reference

- Atoms a appearing in the stream at time points $1, 2, 5$
- Query time $t = 4$. Window on interval $[1, 4]$



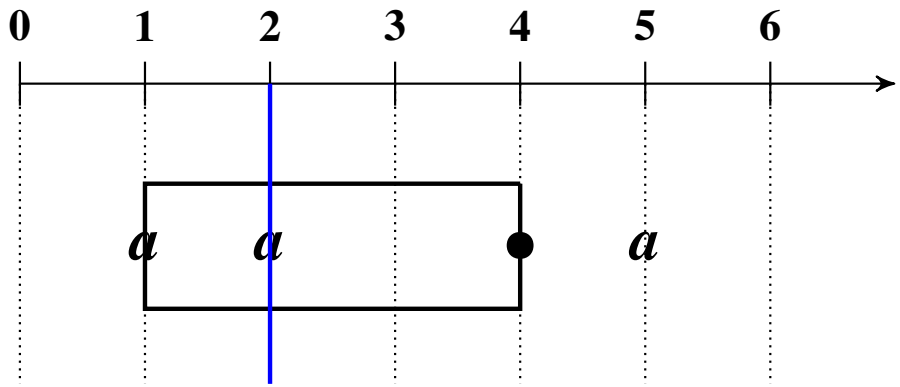
- Example queries: In this window, does a hold. . .

  . . . now, i.e., exactly at t? a no

  . . . at time point 2?

Ideas for Time Reference

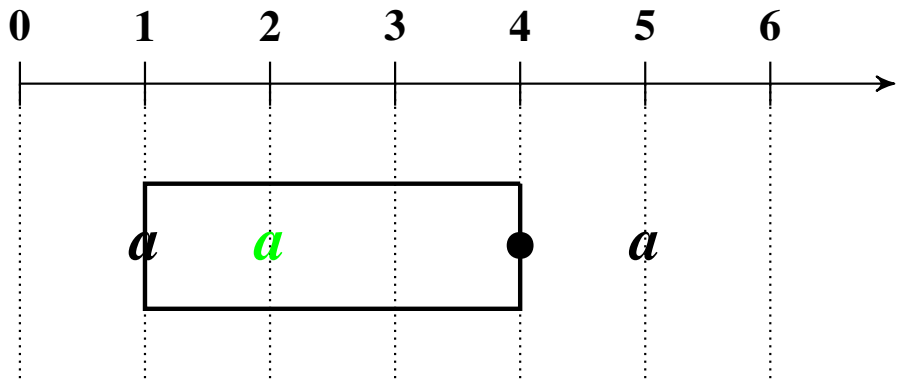
- Atoms a appearing in the stream at time points $1, 2, 5$
- Query time $t = 4$. Window on interval $[1, 4]$



- Example queries: In this window, does a hold...
  ...now, i.e., exactly at t? a no
  ...at time point 2? $@_2\, a$

Ideas for Time Reference

- Atoms a appearing in the stream at time points $1, 2, 5$
- Query time $t = 4$. Window on interval $[1, 4]$



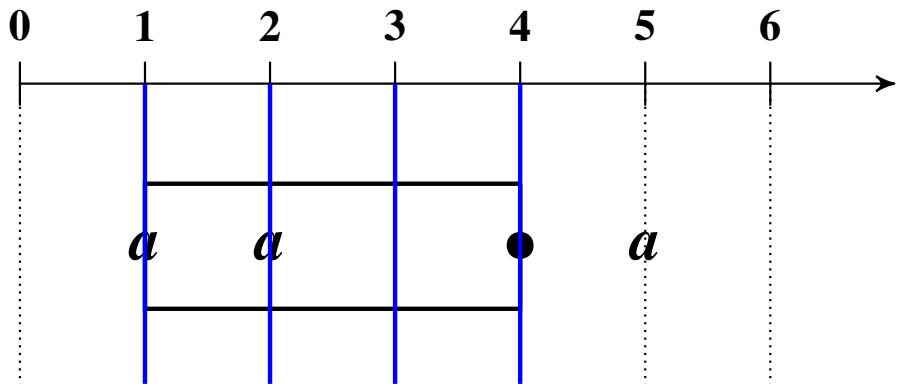
- Example queries: In this window, does a hold. . .

  . . . now, i.e., exactly at t? a no

  . . . at time point 2? $@_2\, a$ yes

Ideas for Time Reference

- Atoms a appearing in the stream at time points $1, 2, 5$
- Query time $t = 4$. Window on interval $[1, 4]$

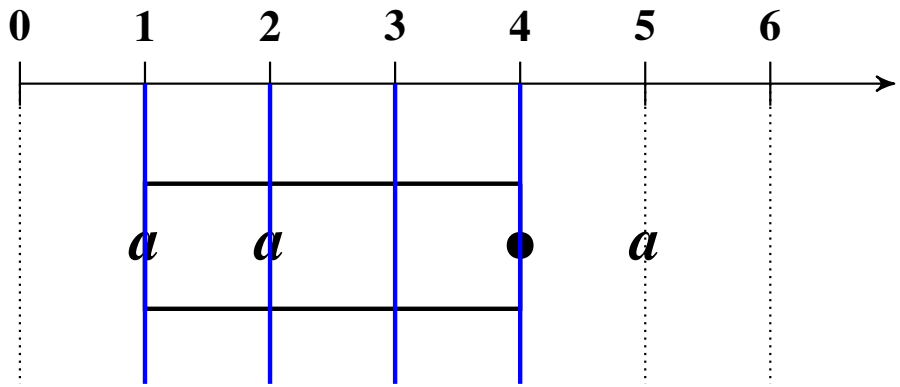


- Example queries: In this window, does a hold...

...now, i.e., exactly at t?	a	no
...at time point 2?	$@_2\, a$	yes
...at some time point t'?		

Ideas for Time Reference

- Atoms a appearing in the stream at time points $1, 2, 5$
- Query time $t = 4$. Window on interval $[1, 4]$

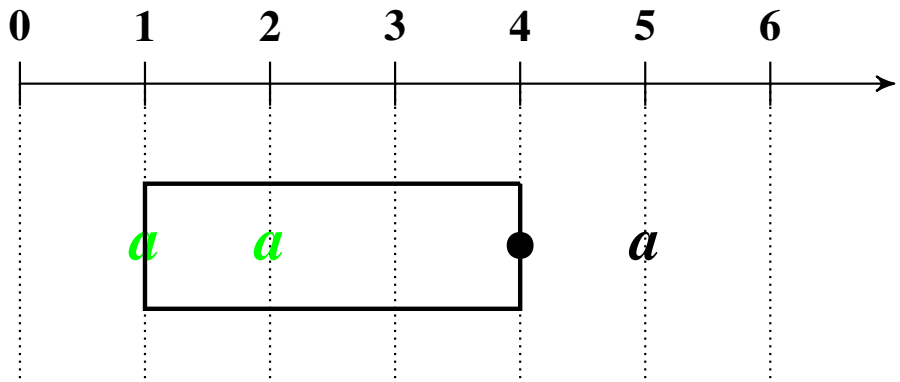


- Example queries: In this window, does a hold...

...now, i.e., exactly at t?	a	no
...at time point 2?	$@_2\, a$	yes
...at some time point t'?	$\Diamond\, a$	

Ideas for Time Reference

- Atoms a appearing in the stream at time points $1, 2, 5$
- Query time $t = 4$. Window on interval $[1, 4]$

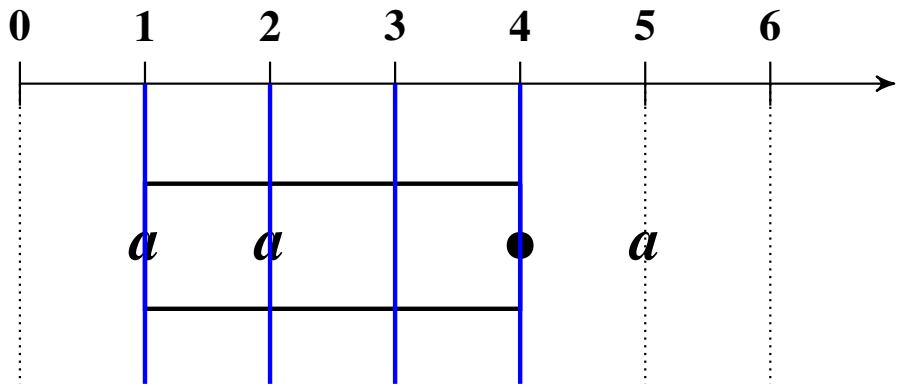


- Example queries: In this window, does a hold...

...now, i.e., exactly at t?	a	no
...at time point 2?	$@_2\, a$	yes
...at some time point t'?	$\Diamond\, a$	yes

Ideas for Time Reference

- Atoms a appearing in the stream at time points $1, 2, 5$
- Query time $t = 4$. Window on interval $[1, 4]$

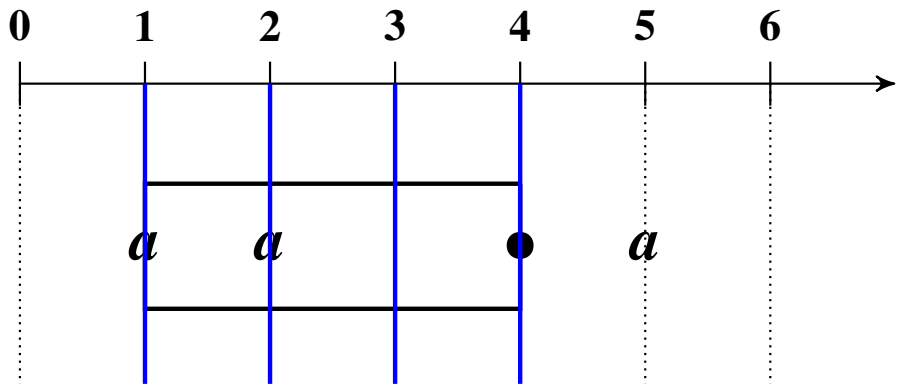


- Example queries: In this window, does a hold...

...now, i.e., exactly at t?	a	no
...at time point 2?	$@_2\, a$	yes
...at some time point t'?	$\Diamond\, a$	yes
...at all time points t'?		

Ideas for Time Reference

- Atoms a appearing in the stream at time points $1, 2, 5$
- Query time $t = 4$. Window on interval $[1, 4]$

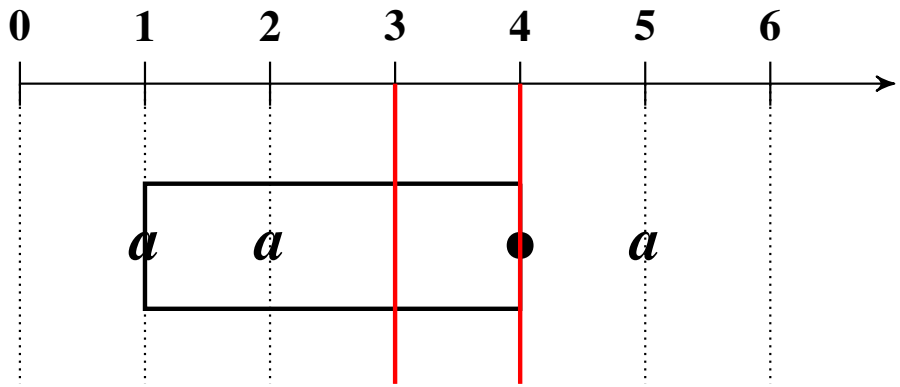


- Example queries: In this window, does a hold...

...now, i.e., exactly at t?	a	no
...at time point 2?	$@_2\, a$	yes
...at some time point t'?	$\Diamond a$	yes
...at all time points t'?	$\Box a$	

Ideas for Time Reference

- Atoms a appearing in the stream at time points $1, 2, 5$
- Query time $t = 4$. Window on interval $[1, 4]$

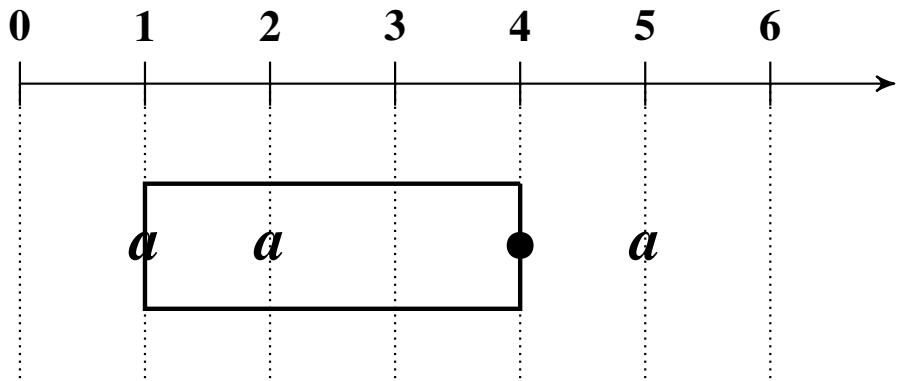


- Example queries: In this window, does a hold...

...now, i.e., exactly at t?	a	no
...at time point 2?	$@_2\, a$	yes
...at some time point t'?	$\Diamond\, a$	yes
...at all time points t'?	$\Box\, a$	no

Ideas for Time Reference

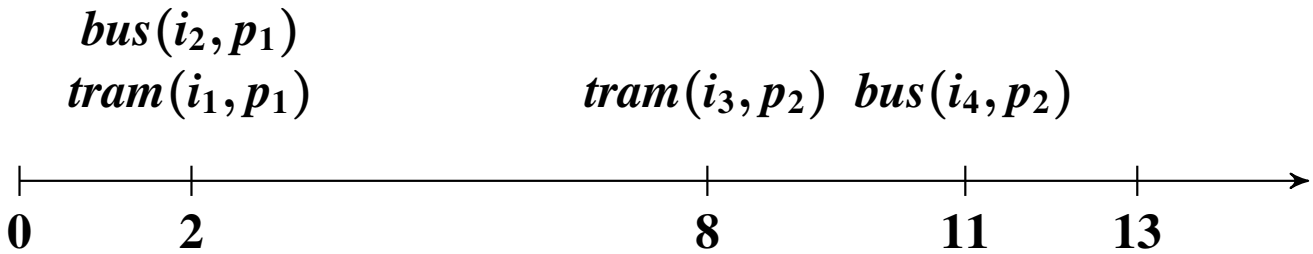
- Atoms a appearing in the stream at time points $1, 2, 5$
- Query time $t = 4$. Window on interval $[1, 4]$



- Example queries: In this window, does a hold...

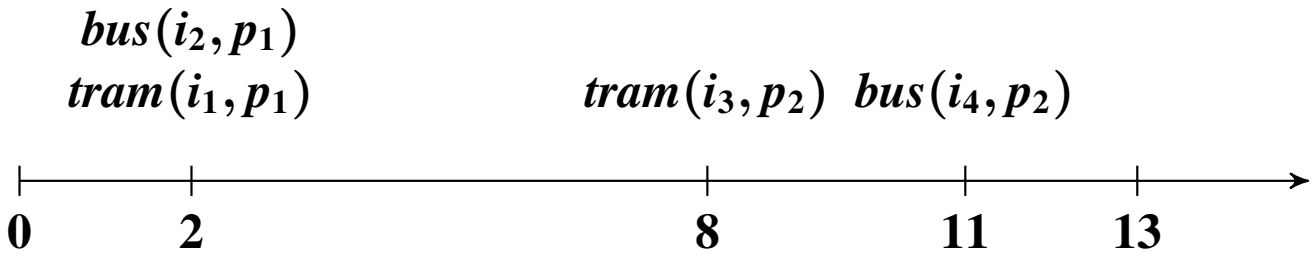
...now, i.e., exactly at t?	a	no
...at time point 2?	$@_2\, a$	yes
...at some time point t'?	$\Diamond\, a$	yes
...at all time points t'?	$\Box\, a$	no

Streams



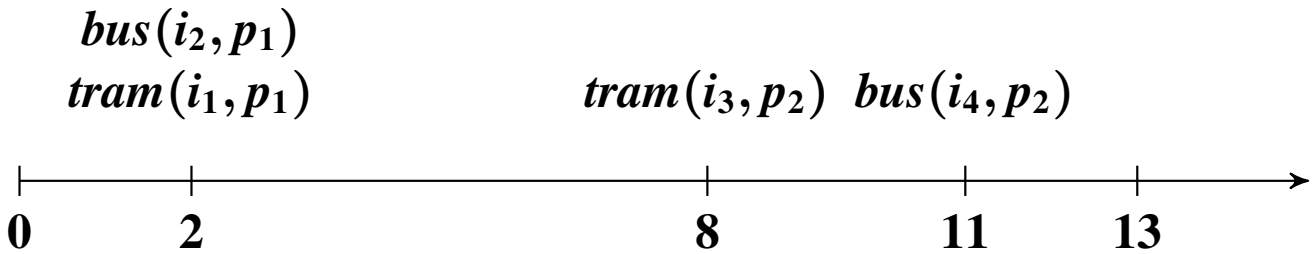
- Stream $S = (T, \upsilon)$, where

Streams



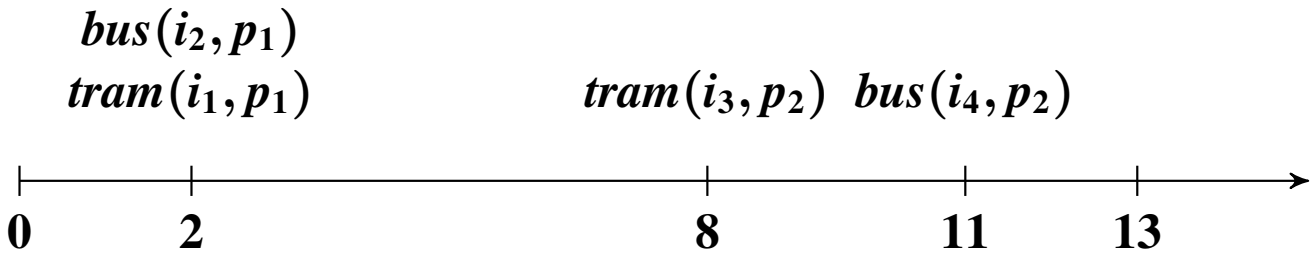
- Stream $S = (T, \upsilon)$, where
  - T: interval in $\mathbb{N}$

Streams



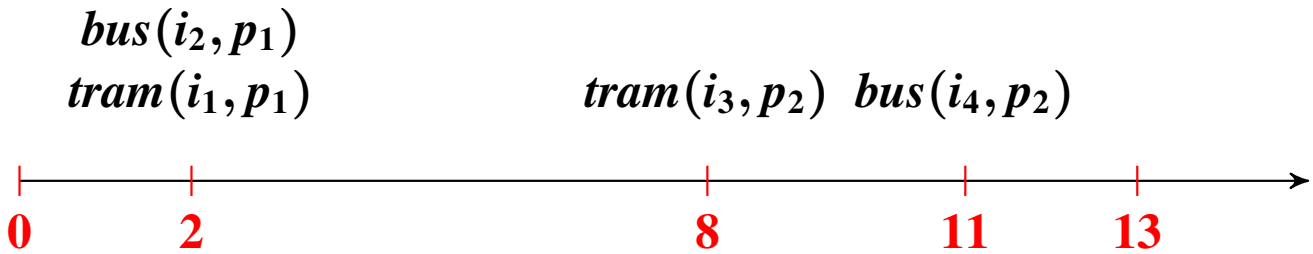
- Stream $S = (T, \upsilon)$, where
  - T: interval in $\mathbb{N}$
  - $\upsilon : T \rightarrow 2^{\mathcal{G}}$ (interpretation of ground atoms $\mathcal{G}$)

Streams



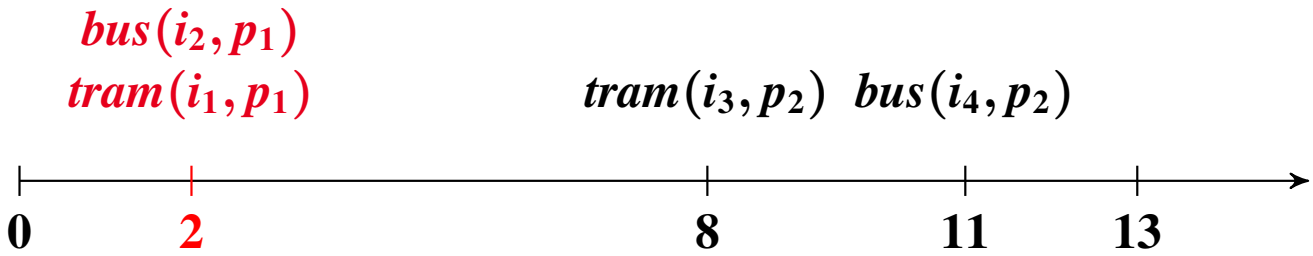
- Stream $S = (T, \upsilon)$, where
  - T: interval in $\mathbb{N}$
  - $\upsilon : T \rightarrow 2^{\mathcal{G}}$ (interpretation of ground atoms $\mathcal{G}$)
- Example

Streams



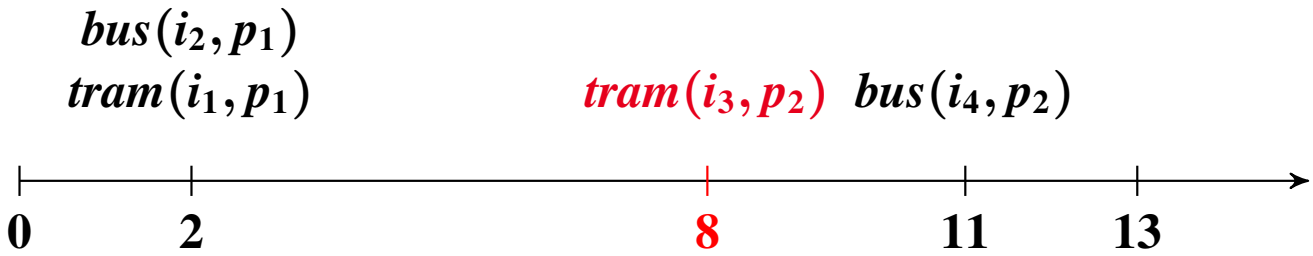
- Stream $S = (T, \upsilon)$, where
  - T: interval in $\mathbb{N}$
  - $\upsilon : T \rightarrow 2^{\mathcal{G}}$ (interpretation of ground atoms $\mathcal{G}$)
- Example
  - $T = [0, 13]$

Streams



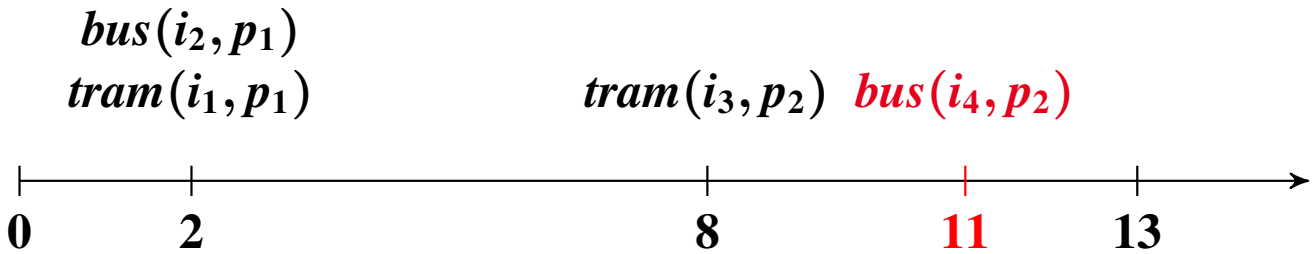
- Stream $S = (T, \upsilon)$, where
  - T: interval in $\mathbb{N}$
  - $\upsilon : T \rightarrow 2^{\mathcal{G}}$ (interpretation of ground atoms $\mathcal{G}$)
- Example
  - $T = [0, 13]$
  - $\upsilon = \left\{ 2 \mapsto \{tram(i_1, p_1), bus(i_2, p_1)\} \right\}$

Streams



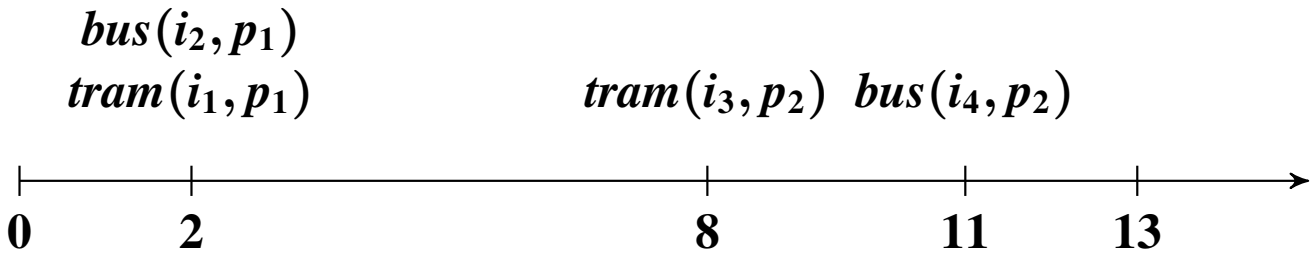
- Stream $S = (T, \upsilon)$, where
  - T: interval in $\mathbb{N}$
  - $\upsilon : T \to 2^{\mathcal{G}}$ (interpretation of ground atoms $\mathcal{G}$)
- Example
  - $T = [0, 13]$
  - $\upsilon = \left\{ 2 \mapsto \{tram(i_1, p_1), bus(i_2, p_1)\},\ 8 \mapsto \{tram(i_3, p_2)\} \right\}$

Streams



- Stream $S = (T, \upsilon)$, where
  - T: interval in $\mathbb{N}$
  - $\upsilon : T \to 2^{\mathcal{G}}$ (interpretation of ground atoms $\mathcal{G}$)
- Example
  - $T = [0, 13]$
  - $\upsilon = \left\{ \begin{array}{l} 2 \mapsto \{tram(i_1, p_1), bus(i_2, p_1)\}, \quad 8 \mapsto \{tram(i_3, p_2)\}, \\ 11 \mapsto \{bus(i_4, p_2)\} \end{array} \right\}$

Streams



- Stream $S = (T, \upsilon)$, where
  - T: interval in $\mathbb{N}$
  - $\upsilon : T \to 2^{\mathcal{G}}$ (interpretation of ground atoms $\mathcal{G}$)
- Example
  - $T = [0, 13]$
  - $\upsilon = \left\{ \begin{array}{l} 2 \mapsto \{tram(i_1, p_1), bus(i_2, p_1)\}, \quad 8 \mapsto \{tram(i_3, p_2)\}, \\ 11 \mapsto \{bus(i_4, p_2)\}, \quad i \mapsto \emptyset \quad \text{else} \end{array} \right\}$

Formulas

- Formulas defined by the grammar (atom a, $t \in \mathbb{N}$ timepoint)

$$\alpha ::=$$

Formulas

- Formulas defined by the grammar (atom a, $t \in \mathbb{N}$ timepoint)

$$\alpha ::= a \mid \neg\alpha \mid \alpha \wedge \alpha \mid \alpha \vee \alpha \mid \alpha \rightarrow \alpha$$

Formulas

- Formulas defined by the grammar (atom a, $t \in \mathbb{N}$ timepoint)

$$\alpha ::= a \mid \neg\alpha \mid \alpha \wedge \alpha \mid \alpha \vee \alpha \mid \alpha \rightarrow \alpha \mid \Diamond\alpha \mid \Box\alpha \mid @_t\alpha$$

Formulas

- Formulas defined by the grammar (atom a, $t \in \mathbb{N}$ timepoint)

$$\alpha ::= a \mid \neg\alpha \mid \alpha \wedge \alpha \mid \alpha \vee \alpha \mid \alpha \rightarrow \alpha \mid \Diamond\alpha \mid \Box\alpha \mid @_t\alpha \mid \boxplus_i \alpha$$

- $\boxplus_i$ window operator: change view on stream

Formulas

- Formulas defined by the grammar (atom a, $t \in \mathbb{N}$ timepoint)

$$\alpha ::= a \mid \neg\alpha \mid \alpha \wedge \alpha \mid \alpha \vee \alpha \mid \alpha \rightarrow \alpha \mid \Diamond\alpha \mid \Box\alpha \mid @_t\alpha \mid \boxplus_i \alpha$$

- $\boxplus_i$ window operator: change view on stream
  - Utilizing window function with identifier i

Formulas

- Formulas defined by the grammar (atom a, $t \in \mathbb{N}$ timepoint)

$$\alpha ::= a \mid \neg\alpha \mid \alpha \wedge \alpha \mid \alpha \vee \alpha \mid \alpha \rightarrow \alpha \mid \Diamond\alpha \mid \Box\alpha \mid @_t\alpha \mid \boxplus_i \alpha$$

- $\boxplus_i$ window operator: change view on stream
  - Utilizing window function with identifier i
  - Change considered substream based on current time point, and
    - current window, or
    - original stream

Formulas

- Formulas defined by the grammar (atom a, $t \in \mathbb{N}$ timepoint)

$$\alpha ::= a \mid \neg\alpha \mid \alpha \wedge \alpha \mid \alpha \vee \alpha \mid \alpha \rightarrow \alpha \mid \Diamond\alpha \mid \Box\alpha \mid @_t\alpha \mid \boxplus_i \alpha$$

- $\boxplus_i$ window operator: change view on stream
  - Utilizing window function with identifier i
  - Change considered substream based on current time point, and
    - current window, or
    - original stream
  - Window operator = window function + stream choice function

Formulas

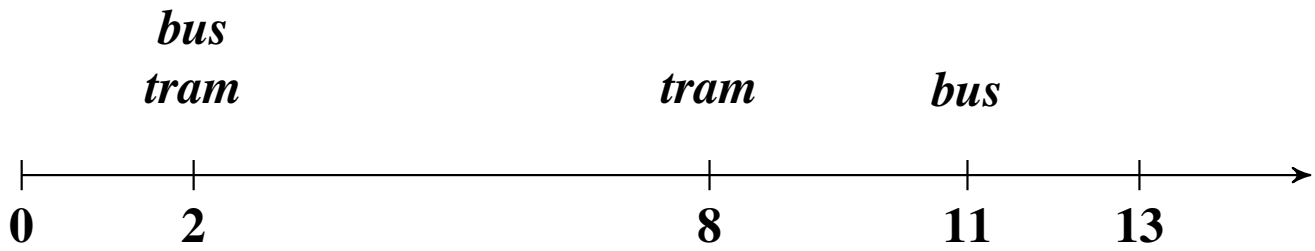
- Formulas defined by the grammar (atom a, $t \in \mathbb{N}$ timepoint)

$$\alpha ::= a \mid \neg\alpha \mid \alpha \wedge \alpha \mid \alpha \vee \alpha \mid \alpha \rightarrow \alpha \mid \Diamond\alpha \mid \Box\alpha \mid @_t\alpha \mid \boxplus_i \alpha$$

- $\boxplus_i$ window operator: change view on stream
  - Utilizing window function with identifier i
  - Change considered substream based on current time point, and
    - current window, or
    - original stream
  - Window operator = window function + stream choice function
  - Why keep the original stream?

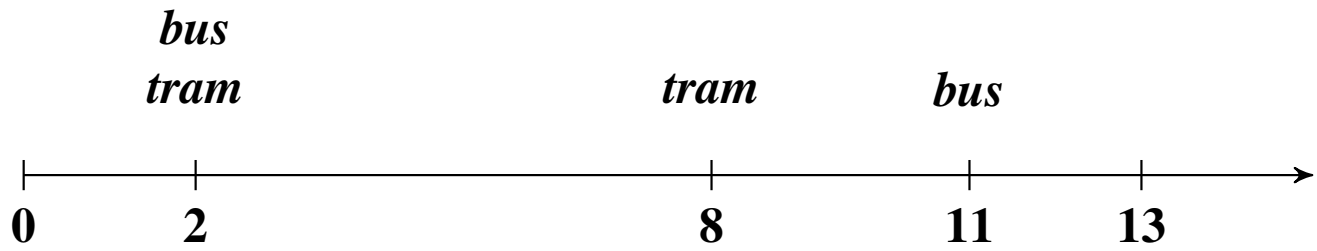
Nested Windows and Stream Choice

- "For the last two trams, did a bus always appear within 5 min?"



Nested Windows and Stream Choice

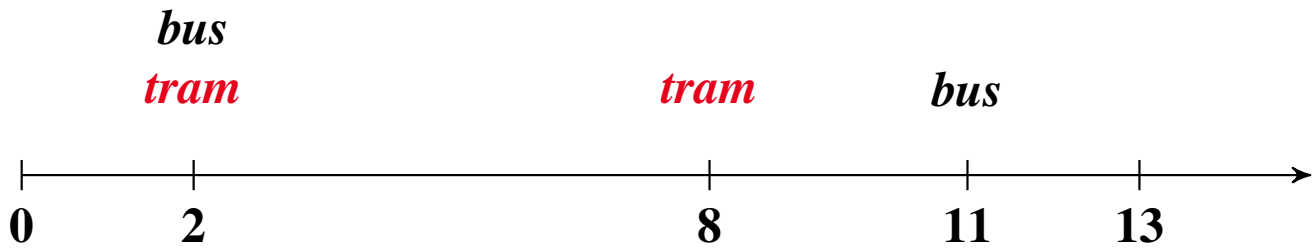
- "For the last two trams, did a bus always appear within 5 min?"



- Partition-based window

Nested Windows and Stream Choice

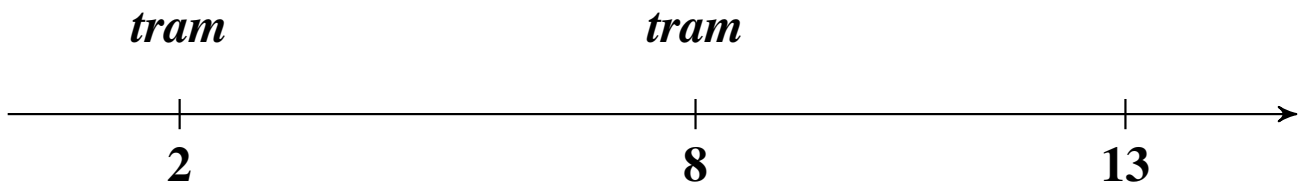
- "For the last two trams, did a bus always appear within 5 min?"



- Partition-based window
  - Partition stream into substreams: trams vs. buses

Nested Windows and Stream Choice

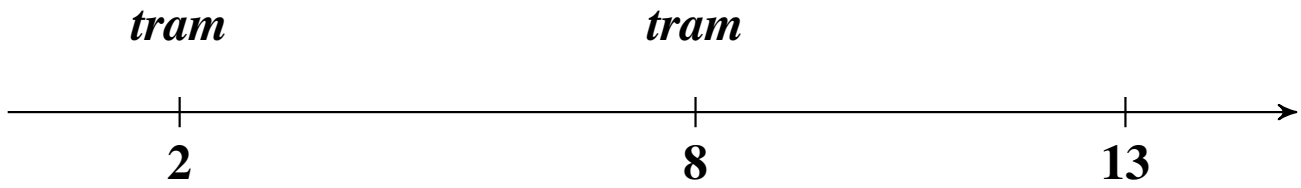
- "For the last two trams, did a bus always appear within 5 min?"



- Partition-based window
  - Partition stream into substreams: trams vs. buses
  - Apply tuple-based windows on substreams: 2 trams, 0 buses

Nested Windows and Stream Choice

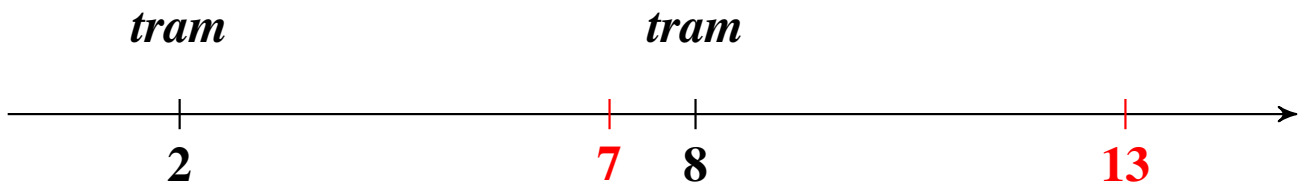
- "For the last two trams, did a bus always appear within 5 min?"



- Partition-based window
  - Partition stream into substreams: trams vs. buses
  - Apply tuple-based windows on substreams: 2 trams, 0 buses
- In the new view, buses are invisible

Nested Windows and Stream Choice

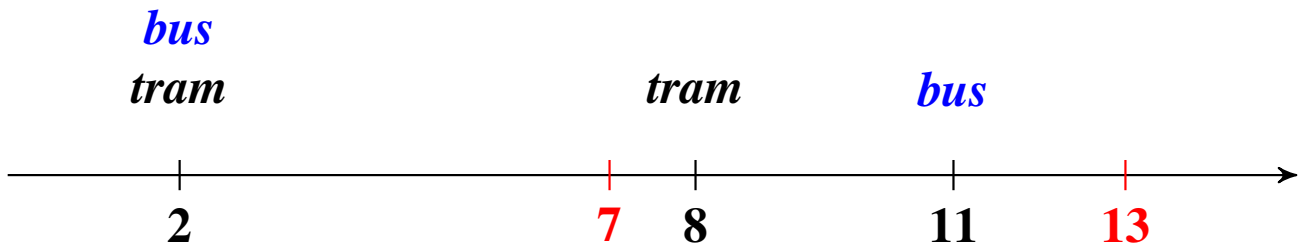
- "For the last two trams, did a bus always appear within 5 min?"



- Partition-based window
  - Partition stream into substreams: trams vs. buses
  - Apply tuple-based windows on substreams: 2 trams, 0 buses
- In the new view, buses are invisible
- $\Rightarrow$ For "within 5 min" window: use data of original stream again

Nested Windows and Stream Choice

- "For the last two trams, did a bus always appear within 5 min?"



- Partition-based window
  - Partition stream into substreams: trams vs. buses
  - Apply tuple-based windows on substreams: 2 trams, 0 buses
- In the new view, buses are invisible
- $\Rightarrow$ For "within 5 min" window: use data of original stream again

Semantics: Structure

- Structure $\boldsymbol{M} = \langle \boldsymbol{T}, \boldsymbol{\upsilon}, \hat{\boldsymbol{W}} \rangle$, where

Semantics: Structure

- Structure $\boldsymbol{M} = \langle \boldsymbol{T}, \boldsymbol{\upsilon}, \hat{\boldsymbol{W}} \rangle$, where
  - $(\boldsymbol{T}, \boldsymbol{\upsilon})$ original stream

Semantics: Structure

- Structure $\boldsymbol{M} = \langle \boldsymbol{T}, \boldsymbol{\upsilon}, \hat{\boldsymbol{W}} \rangle$, where
  - $(\boldsymbol{T}, \boldsymbol{\upsilon})$ original stream
  - $\hat{\boldsymbol{W}}$ mapping from idenfiers (in $\mathbb{N}$) to *extended* window functions

Semantics: Structure

- Structure $\boldsymbol{M} = \langle \boldsymbol{T}, \upsilon, \hat{\boldsymbol{W}} \rangle$, where
  - $(\boldsymbol{T}, \upsilon)$ original stream
  - $\hat{\boldsymbol{W}}$ mapping from idenfiers (in $\mathbb{N}$) to *extended* window functions
  - choice function $\boldsymbol{ch}(\boldsymbol{S_1}, \boldsymbol{S_2}) \mapsto \boldsymbol{S'}$

$$\hat{w}(S_1, S_2, t) = w(ch(S_1, S_2), t)$$

Semantics: Structure

- Structure $\boldsymbol{M} = \langle \boldsymbol{T}, \upsilon, \hat{\boldsymbol{W}} \rangle$, where
  - $(\boldsymbol{T}, \upsilon)$ original stream
  - $\hat{\boldsymbol{W}}$ mapping from idenfiers (in $\mathbb{N}$) to *extended* window functions
  - choice function $\boldsymbol{ch}(\boldsymbol{S_1}, \boldsymbol{S_2}) \mapsto \boldsymbol{S'}$

$$\hat{w}(S_1, S_2, t) = w(ch(S_1, S_2), t)$$

- Example

Semantics: Structure

- Structure $\boldsymbol{M} = \langle \boldsymbol{T}, \upsilon, \hat{\boldsymbol{W}} \rangle$, where
  - $(\boldsymbol{T}, \upsilon)$ original stream
  - $\hat{\boldsymbol{W}}$ mapping from idenfiers (in $\mathbb{N}$) to *extended* window functions
  - choice function $\boldsymbol{ch}(\boldsymbol{S_1}, \boldsymbol{S_2}) \mapsto \boldsymbol{S'}$

$$\hat{w}(S_1, S_2, t) = w(ch(S_1, S_2), t)$$

- Example
  - w^5 time-based window for last 5 minutes

Semantics: Structure

- Structure $\boldsymbol{M} = \langle \boldsymbol{T}, \upsilon, \hat{\boldsymbol{W}} \rangle$, where
  - $(\boldsymbol{T}, \upsilon)$ original stream
  - $\hat{\boldsymbol{W}}$ mapping from idenfiers (in $\mathbb{N}$) to *extended* window functions
  - choice function $\boldsymbol{ch}(\boldsymbol{S_1}, \boldsymbol{S_2}) \mapsto \boldsymbol{S'}$

$$\hat{w}(S_1, S_2, t) = w(ch(S_1, S_2), t)$$

- Example
  - $\boldsymbol{w^5}$ time-based window for last 5 minutes
  - $\boldsymbol{ch_2}$ choice that selects the second stream $(\boldsymbol{ch_2}(\boldsymbol{S_1}, \boldsymbol{S_2}) = \boldsymbol{S_2})$

Semantics: Structure

- Structure $M = \langle T, \upsilon, \hat{W} \rangle$, where
  - (T, υ) original stream
  - $\hat{W}$ mapping from idenfiers (in $\mathbb{N}$) to *extended* window functions
  - choice function $ch(S_1, S_2) \mapsto S'$

$$\hat{w}(S_1, S_2, t) = w(ch(S_1, S_2), t)$$

- Example
  - w^5 time-based window for last 5 minutes
  - ch_2 choice that selects the second stream $(ch_2(S_1, S_2) = S_2)$
  - $\hat{W}(1) = \hat{w}^5$, where $\hat{w}^5(S_1, S_2, t) = w^5(S_2, t)$

Semantics: Entailment

- Structure $\boldsymbol{M} = \langle \boldsymbol{T}, \upsilon, \hat{\boldsymbol{W}} \rangle$ with original stream $\boldsymbol{S_M} = (\boldsymbol{T}, \upsilon)$

Semantics: Entailment

- Structure $\boldsymbol{M} = \langle \boldsymbol{T}, \upsilon, \hat{\boldsymbol{W}} \rangle$ with original stream $\boldsymbol{S_M} = (\boldsymbol{T}, \upsilon)$
- Substream $\boldsymbol{S} = (\boldsymbol{T_S}, \upsilon_S)$ of $\boldsymbol{S_M}$: currently considered window

Semantics: Entailment

- Structure $\boldsymbol{M} = \langle \boldsymbol{T}, \upsilon, \hat{\boldsymbol{W}} \rangle$ with original stream $\boldsymbol{S_M} = (\boldsymbol{T}, \upsilon)$
- Substream $\boldsymbol{S} = (\boldsymbol{T_S}, \upsilon_S)$ of $\boldsymbol{S_M}$: currently considered window
- Time point $\boldsymbol{t} \in \boldsymbol{T_S}$ (query time)

Semantics: Entailment

- Structure $\boldsymbol{M} = \langle \boldsymbol{T}, \upsilon, \hat{\boldsymbol{W}} \rangle$ with original stream $\boldsymbol{S_M} = (\boldsymbol{T}, \upsilon)$
- Substream $\boldsymbol{S} = (\boldsymbol{T_S}, \upsilon_S)$ of $\boldsymbol{S_M}$: currently considered window
- Time point $\boldsymbol{t} \in \boldsymbol{T_S}$ (query time)
- Entailment between $\boldsymbol{M}, \boldsymbol{S}, \boldsymbol{t}$ and formulas α, β

Semantics: Entailment

- Structure $\boldsymbol{M} = \langle \boldsymbol{T}, \upsilon, \hat{\boldsymbol{W}} \rangle$ with original stream $\boldsymbol{S_M} = (\boldsymbol{T}, \upsilon)$
- Substream $\boldsymbol{S} = (\boldsymbol{T_S}, \upsilon_S)$ of $\boldsymbol{S_M}$: currently considered window
- Time point $\boldsymbol{t} \in \boldsymbol{T_S}$ (query time)
- Entailment between $\boldsymbol{M}, \boldsymbol{S}, \boldsymbol{t}$ and formulas α, β

$$\boldsymbol{M}, \boldsymbol{S}, \boldsymbol{t} \Vdash \boldsymbol{a} \qquad \text{iff} \quad \boldsymbol{a} \in \upsilon_S(\boldsymbol{t}) \,,$$

Semantics: Entailment

- Structure $M = \langle T, v, \hat{W} \rangle$ with original stream $S_M = (T, v)$
- Substream $S = (T_S, v_S)$ of S_M: currently considered window
- Time point $t \in T_S$ (query time)
- Entailment between M, S, t and formulas α, β

$$
\begin{array}{lll}
M, S, t \Vdash a & \text{iff} & a \in v_S(t)\,, \\
M, S, t \Vdash \neg\alpha & \text{iff} & M, S, t \nVdash \alpha, \\
M, S, t \Vdash \alpha \wedge \beta & \text{iff} & M, S, t \Vdash \alpha \text{ and } M, S, t \Vdash \beta, \\
M, S, t \Vdash \alpha \vee \beta & \text{iff} & M, S, t \Vdash \alpha \text{ or } M, S, t \Vdash \beta, \\
M, S, t \Vdash \alpha \rightarrow \beta & \text{iff} & M, S, t \nVdash \alpha \text{ or } M, S, t \Vdash \beta,
\end{array}
$$

Semantics: Entailment

- Structure $M = \langle T, v, \hat{W} \rangle$ with original stream $S_M = (T, v)$
- Substream $S = (T_S, v_S)$ of S_M: currently considered window
- Time point $t \in T_S$ (query time)
- Entailment between M, S, t and formulas α, β

$$
\begin{array}{lll}
M, S, t \Vdash a & \text{iff} & a \in v_S(t)\,, \\
M, S, t \Vdash \neg\alpha & \text{iff} & M, S, t \nVdash \alpha, \\
M, S, t \Vdash \alpha \wedge \beta & \text{iff} & M, S, t \Vdash \alpha \text{ and } M, S, t \Vdash \beta, \\
M, S, t \Vdash \alpha \vee \beta & \text{iff} & M, S, t \Vdash \alpha \text{ or } M, S, t \Vdash \beta, \\
M, S, t \Vdash \alpha \rightarrow \beta & \text{iff} & M, S, t \nVdash \alpha \text{ or } M, S, t \Vdash \beta, \\
M, S, t \Vdash \Diamond\alpha & \text{iff} & M, S, t' \Vdash \alpha \text{ for some } t' \in T_S,
\end{array}
$$

Semantics: Entailment

- Structure $\boldsymbol{M} = \langle \boldsymbol{T}, \upsilon, \hat{\boldsymbol{W}} \rangle$ with original stream $\boldsymbol{S_M} = (\boldsymbol{T}, \upsilon)$
- Substream $\boldsymbol{S} = (\boldsymbol{T_S}, \upsilon_S)$ of $\boldsymbol{S_M}$: currently considered window
- Time point $\boldsymbol{t} \in \boldsymbol{T_S}$ (query time)
- Entailment between $\boldsymbol{M}, \boldsymbol{S}, \boldsymbol{t}$ and formulas α, β

$$
\begin{array}{lll}
\boldsymbol{M}, \boldsymbol{S}, \boldsymbol{t} \Vdash \boldsymbol{a} & \text{iff} & \boldsymbol{a} \in \upsilon_S(\boldsymbol{t}) \,, \\
\boldsymbol{M}, \boldsymbol{S}, \boldsymbol{t} \Vdash \neg\alpha & \text{iff} & \boldsymbol{M}, \boldsymbol{S}, \boldsymbol{t} \nVdash \alpha, \\
\boldsymbol{M}, \boldsymbol{S}, \boldsymbol{t} \Vdash \alpha \wedge \beta & \text{iff} & \boldsymbol{M}, \boldsymbol{S}, \boldsymbol{t} \Vdash \alpha \text{ and } \boldsymbol{M}, \boldsymbol{S}, \boldsymbol{t} \Vdash \beta, \\
\boldsymbol{M}, \boldsymbol{S}, \boldsymbol{t} \Vdash \alpha \vee \beta & \text{iff} & \boldsymbol{M}, \boldsymbol{S}, \boldsymbol{t} \Vdash \alpha \text{ or } \boldsymbol{M}, \boldsymbol{S}, \boldsymbol{t} \Vdash \beta, \\
\boldsymbol{M}, \boldsymbol{S}, \boldsymbol{t} \Vdash \alpha \rightarrow \beta & \text{iff} & \boldsymbol{M}, \boldsymbol{S}, \boldsymbol{t} \nVdash \alpha \text{ or } \boldsymbol{M}, \boldsymbol{S}, \boldsymbol{t} \Vdash \beta, \\
\boldsymbol{M}, \boldsymbol{S}, \boldsymbol{t} \Vdash \Diamond\alpha & \text{iff} & \boldsymbol{M}, \boldsymbol{S}, \boldsymbol{t'} \Vdash \alpha \text{ for some } \boldsymbol{t'} \in \boldsymbol{T_S}, \\
\boldsymbol{M}, \boldsymbol{S}, \boldsymbol{t} \Vdash \Box\alpha & \text{iff} & \boldsymbol{M}, \boldsymbol{S}, \boldsymbol{t'} \Vdash \alpha \text{ for all } \boldsymbol{t'} \in \boldsymbol{T_S} \,,
\end{array}
$$

Semantics: Entailment

- Structure $\boldsymbol{M} = \langle \boldsymbol{T}, \upsilon, \hat{\boldsymbol{W}} \rangle$ with original stream $\boldsymbol{S_M} = (\boldsymbol{T}, \upsilon)$
- Substream $\boldsymbol{S} = (\boldsymbol{T_S}, \upsilon_S)$ of $\boldsymbol{S_M}$: currently considered window
- Time point $\boldsymbol{t} \in \boldsymbol{T_S}$ (query time)
- Entailment between $\boldsymbol{M}, \boldsymbol{S}, \boldsymbol{t}$ and formulas α, β

$$
\begin{array}{lll}
\boldsymbol{M},\boldsymbol{S},\boldsymbol{t} \Vdash \boldsymbol{a} & \text{iff} & \boldsymbol{a} \in \upsilon_S(\boldsymbol{t})\,, \\
\boldsymbol{M},\boldsymbol{S},\boldsymbol{t} \Vdash \neg\alpha & \text{iff} & \boldsymbol{M},\boldsymbol{S},\boldsymbol{t} \nVdash \alpha, \\
\boldsymbol{M},\boldsymbol{S},\boldsymbol{t} \Vdash \alpha \wedge \beta & \text{iff} & \boldsymbol{M},\boldsymbol{S},\boldsymbol{t} \Vdash \alpha \text{ and } \boldsymbol{M},\boldsymbol{S},\boldsymbol{t} \Vdash \beta, \\
\boldsymbol{M},\boldsymbol{S},\boldsymbol{t} \Vdash \alpha \vee \beta & \text{iff} & \boldsymbol{M},\boldsymbol{S},\boldsymbol{t} \Vdash \alpha \text{ or } \boldsymbol{M},\boldsymbol{S},\boldsymbol{t} \Vdash \beta, \\
\boldsymbol{M},\boldsymbol{S},\boldsymbol{t} \Vdash \alpha \rightarrow \beta & \text{iff} & \boldsymbol{M},\boldsymbol{S},\boldsymbol{t} \nVdash \alpha \text{ or } \boldsymbol{M},\boldsymbol{S},\boldsymbol{t} \Vdash \beta, \\
\boldsymbol{M},\boldsymbol{S},\boldsymbol{t} \Vdash \Diamond\alpha & \text{iff} & \boldsymbol{M},\boldsymbol{S},\boldsymbol{t'} \Vdash \alpha \text{ for some } \boldsymbol{t'} \in \boldsymbol{T_S}, \\
\boldsymbol{M},\boldsymbol{S},\boldsymbol{t} \Vdash \Box\alpha & \text{iff} & \boldsymbol{M},\boldsymbol{S},\boldsymbol{t'} \Vdash \alpha \text{ for all } \boldsymbol{t'} \in \boldsymbol{T_S}\,, \\
\boldsymbol{M},\boldsymbol{S},\boldsymbol{t} \Vdash @_{t'}\alpha & \text{iff} & \boldsymbol{M},\boldsymbol{S},\boldsymbol{t'} \Vdash \alpha \text{ and } \boldsymbol{t'} \in \boldsymbol{T_S}\,,
\end{array}
$$

Semantics: Entailment

- Structure $M = \langle T, v, \hat{W} \rangle$ with original stream $S_M = (T, v)$
- Substream $S = (T_S, v_S)$ of S_M: currently considered window
- Time point $t \in T_S$ (query time)
- Entailment between M, S, t and formulas α, β

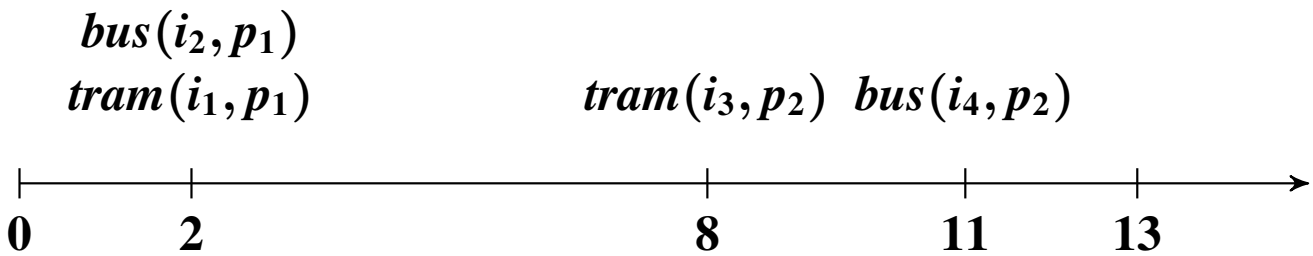
$$
\begin{array}{lll}
M, S, t \Vdash a & \text{iff} & a \in v_S(t)\,, \\
M, S, t \Vdash \neg\alpha & \text{iff} & M, S, t \nVdash \alpha, \\
M, S, t \Vdash \alpha \wedge \beta & \text{iff} & M, S, t \Vdash \alpha \text{ and } M, S, t \Vdash \beta, \\
M, S, t \Vdash \alpha \vee \beta & \text{iff} & M, S, t \Vdash \alpha \text{ or } M, S, t \Vdash \beta, \\
M, S, t \Vdash \alpha \rightarrow \beta & \text{iff} & M, S, t \nVdash \alpha \text{ or } M, S, t \Vdash \beta, \\
M, S, t \Vdash \Diamond\alpha & \text{iff} & M, S, t' \Vdash \alpha \text{ for some } t' \in T_S, \\
M, S, t \Vdash \Box\alpha & \text{iff} & M, S, t' \Vdash \alpha \text{ for all } t' \in T_S\,, \\
M, S, t \Vdash @_{t'}\alpha & \text{iff} & M, S, t' \Vdash \alpha \text{ and } t' \in T_S\,, \\
M, S, t \Vdash \boxplus_i\alpha & \text{iff} & M, S', t \Vdash \alpha \text{ where } S' = \hat{w}_i(S_M, S, t).
\end{array}
$$

Queries

- Query $\alpha[t]$: "$M, S_M, t \Vdash \alpha$"?

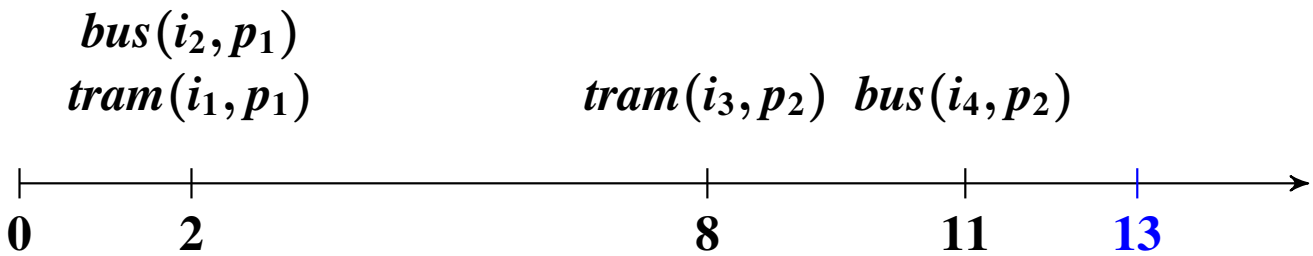
Queries

- Query $\alpha[t]$: "$M, S_M, t \Vdash \alpha$"?



Queries

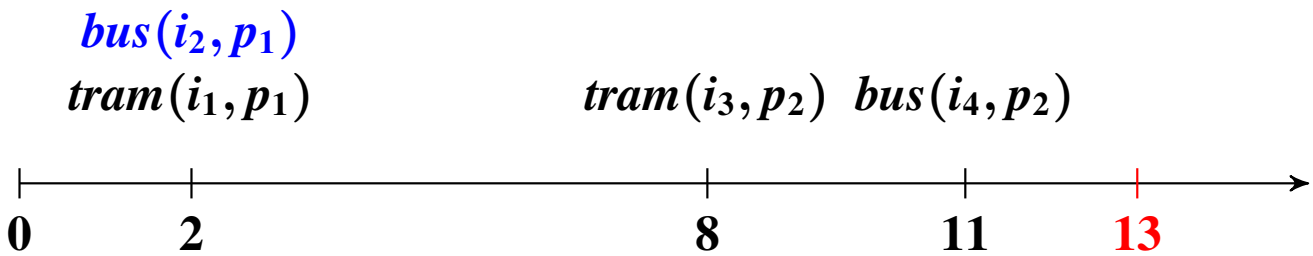
- Query $\alpha[t]$: "$M, S_M, t \Vdash \alpha$"?



$$M, S_M, 13 \Vdash bus(i_2, p_1)?$$

Queries

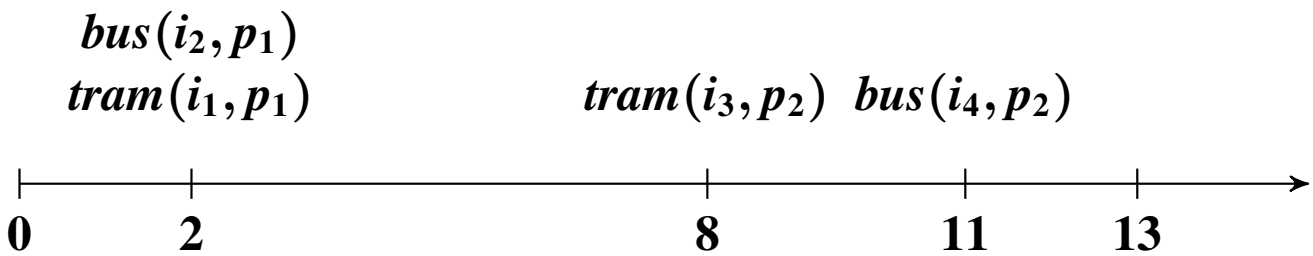
- Query $\alpha[t]$: "$M, S_M, t \Vdash \alpha$"?



$M, S_M, 13 \nVdash bus(i_2, p_1)$, since $bus(i_2, p_1) \notin \upsilon(13)$

Queries

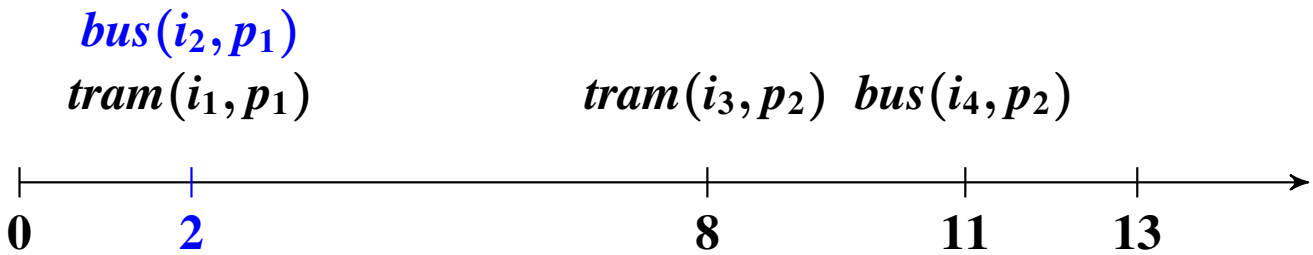
- Query $\alpha[t]$: "$M, S_M, t \Vdash \alpha$"?



$$M, S_M, 13 \Vdash \Diamond bus(i_2, p_1)?$$

Queries

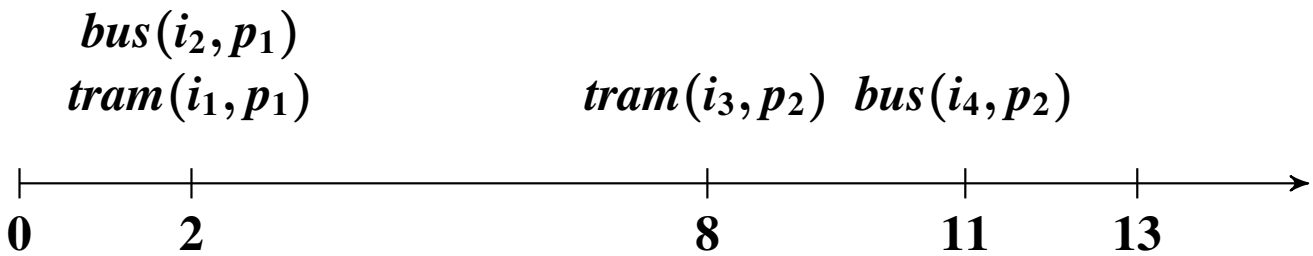
- Query $\alpha[t]$: "$M, S_M, t \Vdash \alpha$"?



$M, S_M, 13 \Vdash \Diamond bus(i_2, p_1)$, since $\exists t' \in T_{S_M}$ s.t. $bus(i_2, p_1) \in \upsilon(t')$

Queries

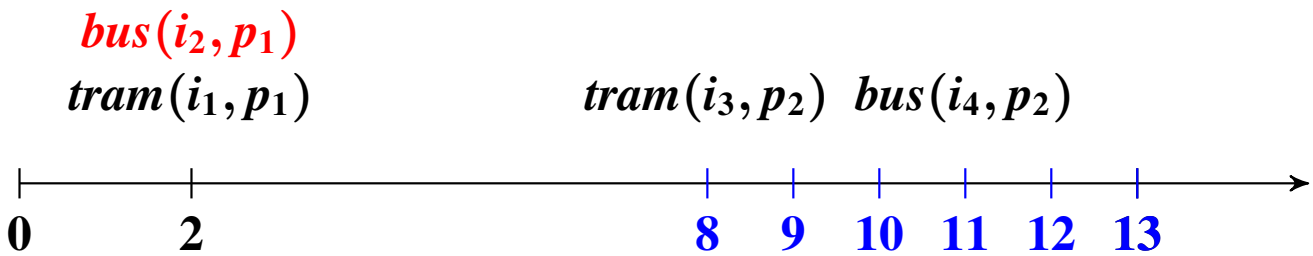
- Query $\alpha[t]$: "$M, S_M, t \Vdash \alpha$"? $\boxplus_1$: last 5 min



$$M, S_M, 13 \Vdash \boxplus_1 \Diamond\, bus(i_2, p_1)?$$

Queries

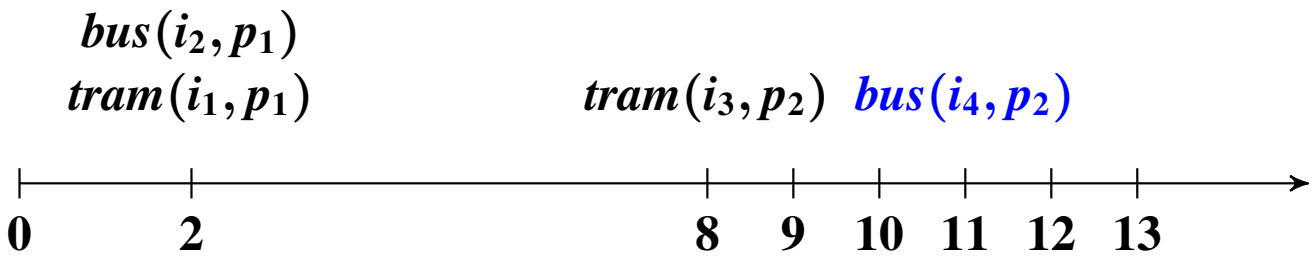
- Query $\alpha[t]$: "$M, S_M, t \Vdash \alpha$"? $\boxplus_1$: last 5 min



$$M, S_M, 13 \not\Vdash \boxplus_1 \Diamond \, bus(i_2, p_1)$$

Queries

- Query $\alpha[t]$: "$M, S_M, t \Vdash \alpha$"? $\boxplus_1$: last 5 min



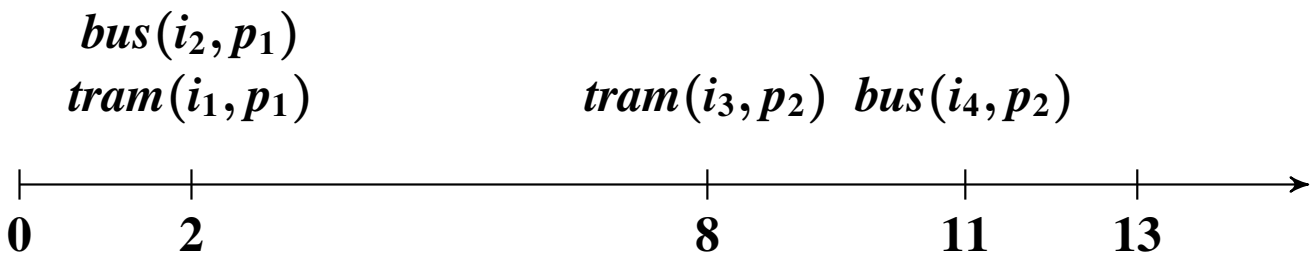
$$M, S_M, 13 \Vdash \boxplus_1 \Diamond \, bus(i_4, p_2)$$

Non-ground Queries

- Non-ground query: Assignments s.t. substitution hold

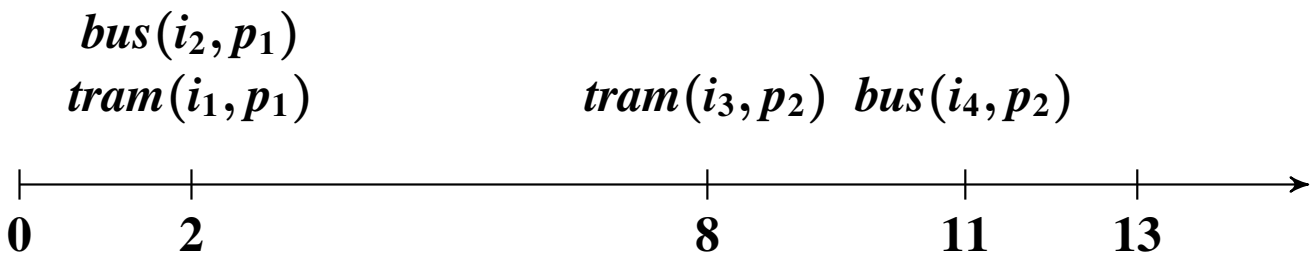
Non-ground Queries

- Non-ground query: Assignments s.t. substitution hold



Non-ground Queries

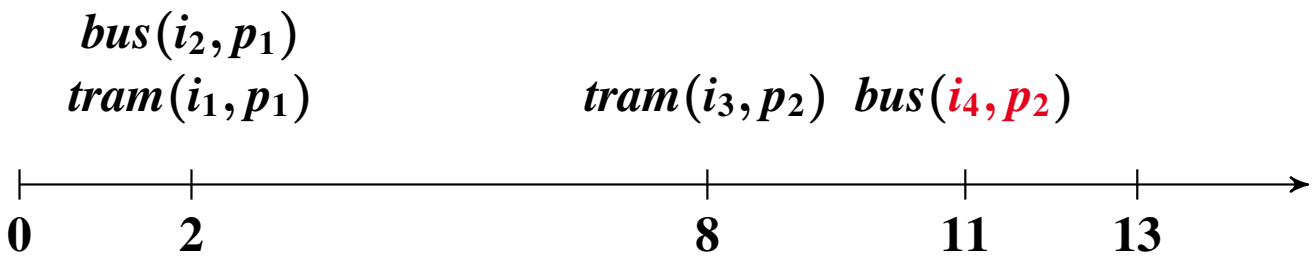
- Non-ground query: Assignments s.t. substitution hold



$$M, S_M, 13 \Vdash \boxplus_1 \Diamond bus(X, P)?$$

Non-ground Queries

- Non-ground query: Assignments s.t. substitution hold

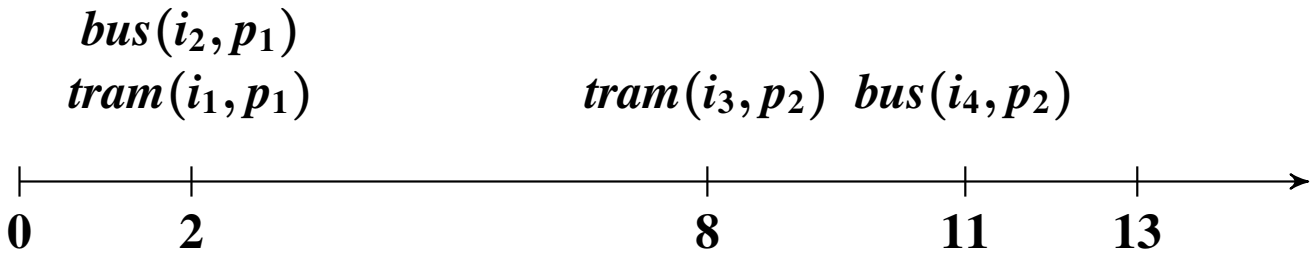


$$M, S_M, 13 \Vdash \boxplus_1 \Diamond bus(X, P)?$$

$$X \mapsto i_4,\ P \mapsto p_2$$

Non-ground Queries

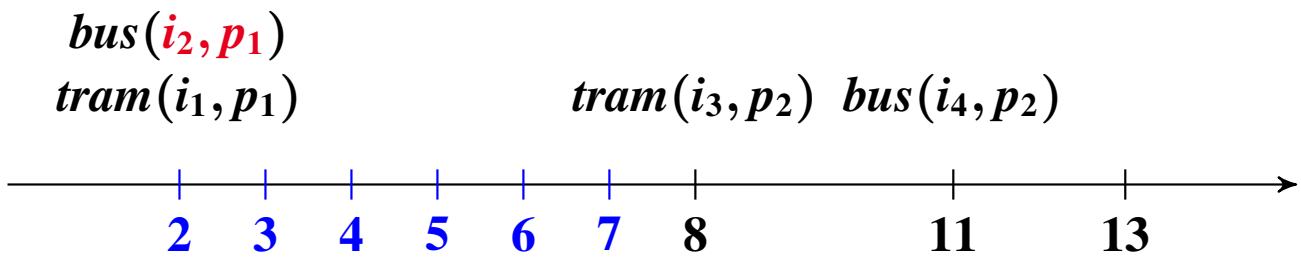
- Non-ground query: Assignments s.t. substitution hold



$$M, S, U \Vdash \boxplus_1 \Diamond bus(i_2, p_1)?$$

Non-ground Queries

- Non-ground query: Assignments s.t. substitution hold

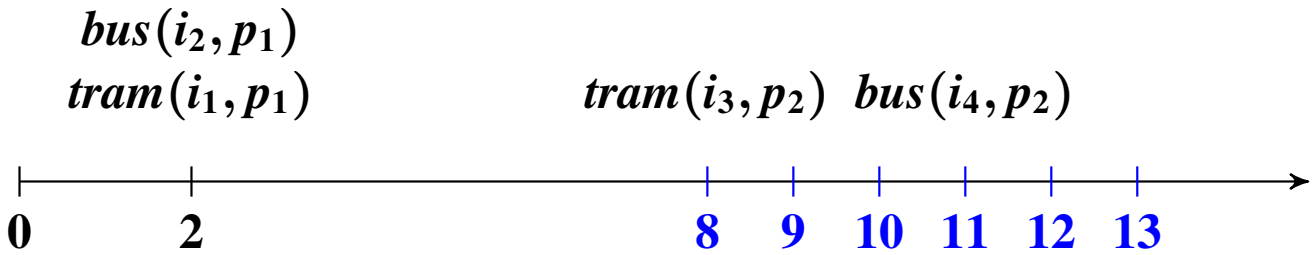


$$M, S, U \Vdash \boxplus_1 \Diamond bus(i_2, p_1)?$$

$$U \mapsto 2, \ldots, 7$$

Non-ground Queries

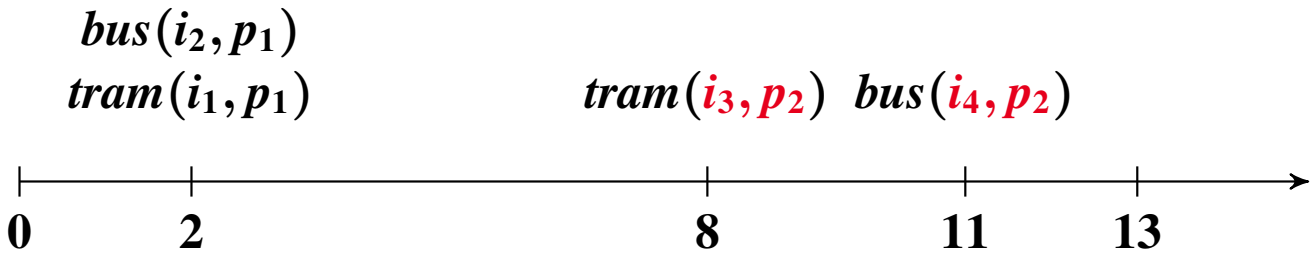
- Non-ground query: Assignments s.t. substitution hold



$$M, S_M, 13 \Vdash \boxplus_1(\Diamond tram(X, P) \wedge \Diamond bus(Y, P))?$$

Non-ground Queries

- Non-ground query: Assignments s.t. substitution hold

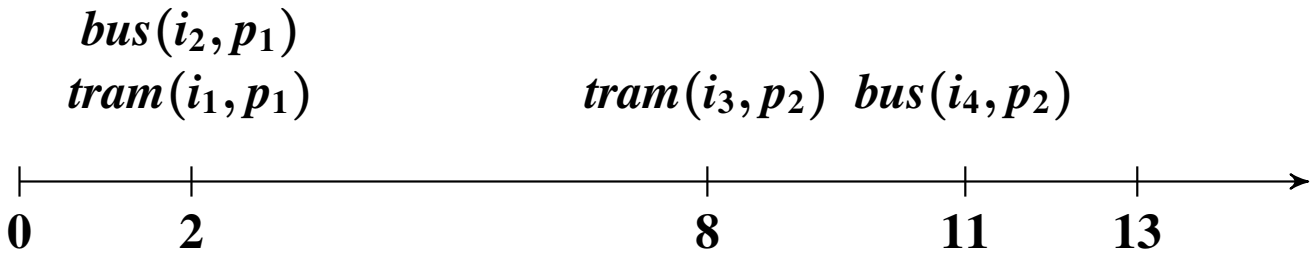


$$M, S_M, 13 \Vdash \boxplus_1 (\Diamond tram(X, P) \wedge \Diamond bus(Y, P))?$$

$$X \mapsto i_3,\ P \mapsto p_2, Y \mapsto i_4$$

Non-ground Queries

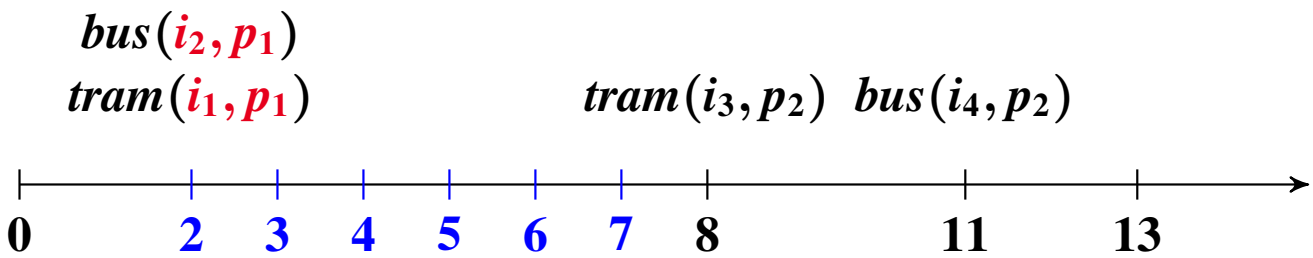
- Non-ground query: Assignments s.t. substitution hold



$$M, S_M, \boldsymbol{U} \Vdash \boxplus_1 \Diamond (tram(X, P) \wedge bus(Y, P))?$$

Non-ground Queries

- Non-ground query: Assignments s.t. substitution hold

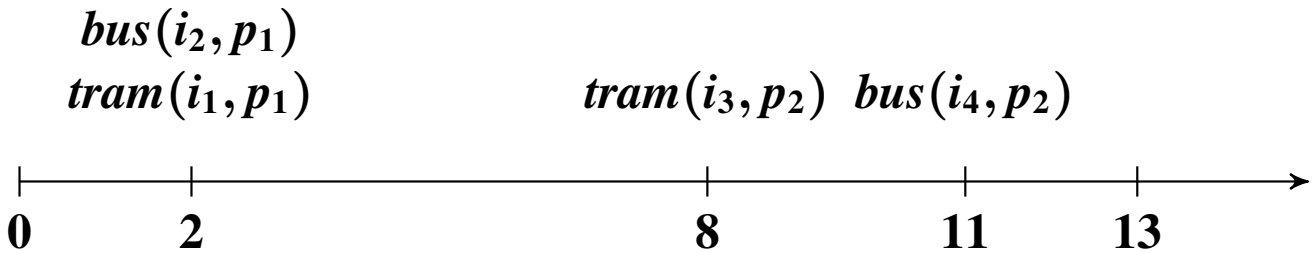


$$M, S_M, U \Vdash \boxplus_1 \Diamond (tram(X, P) \wedge bus(Y, P))?$$

$$U \mapsto 2, \ldots, 7 \quad \times \quad X \mapsto i_1,\ P \mapsto p_1, Y \mapsto i_2$$

Non-ground Queries

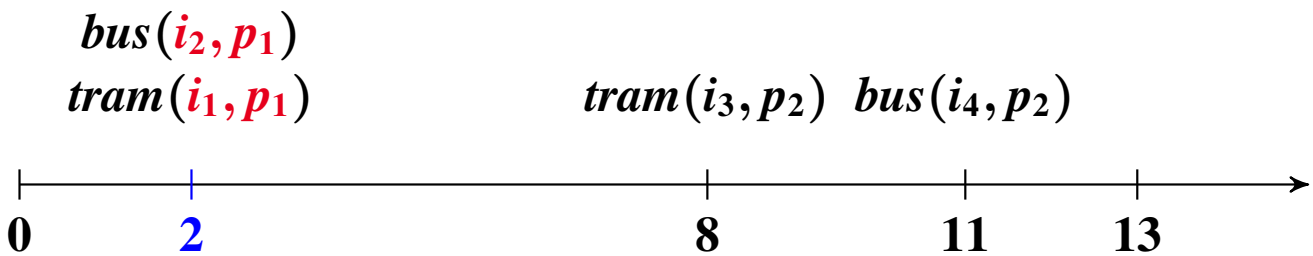
- Non-ground query: Assignments s.t. substitution hold



$$M, S_M, 13 \Vdash @_U(tram(X, P)) \wedge bus(Y, P))?$$

Non-ground Queries

- Non-ground query: Assignments s.t. substitution hold

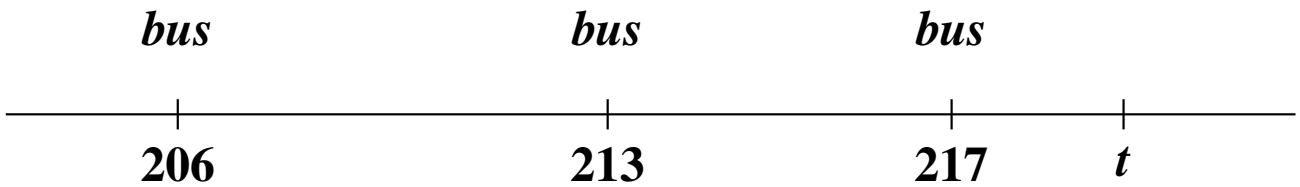


$$M, S_M, 13 \Vdash @_U(tram(X, P)) \wedge bus(Y, P))?$$

$$U \mapsto 2, \qquad X \mapsto i_1,\ P \mapsto p_1, Y \mapsto i_2$$

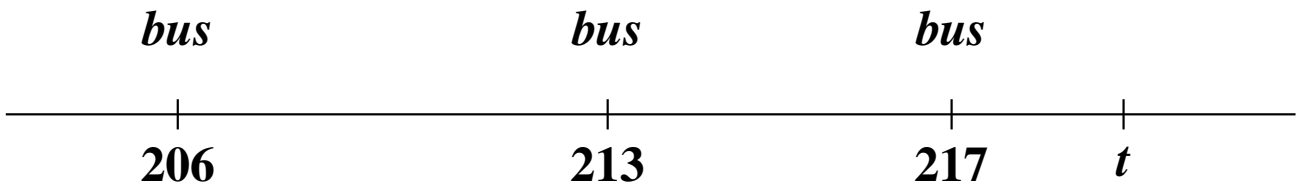
Example: Nested Window

- “In the last hour, did a bus always appear in the last 5 minutes?”



Example: Nested Window

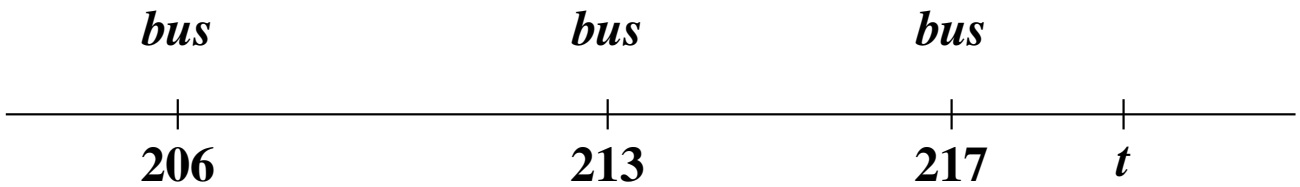
- "In the last hour, did a bus always appear in the last 5 minutes?"



- $\boxplus_i$: time-based window for last i minutes

Example: Nested Window

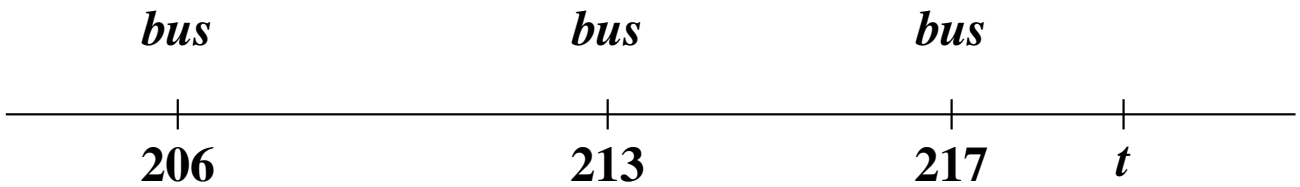
- "In the last hour, did a bus always appear in the last 5 minutes?"



- $\boxplus_i$: time-based window for last i minutes
- Query: $\boxplus_{60}$

Example: Nested Window

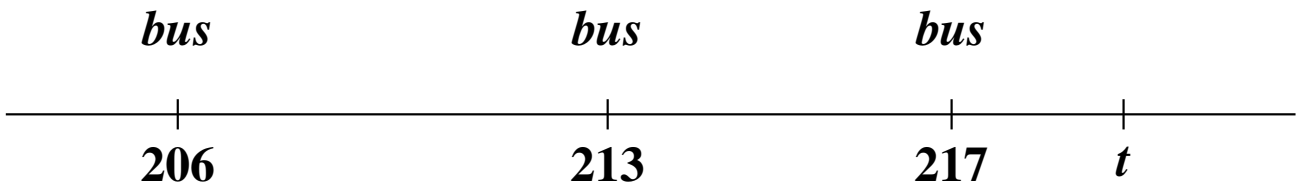
- "In the last hour, did a bus always appear in the last 5 minutes?"



- $\boxplus_i$: time-based window for last i minutes
- Query: $\boxplus_{60} \Box$

Example: Nested Window

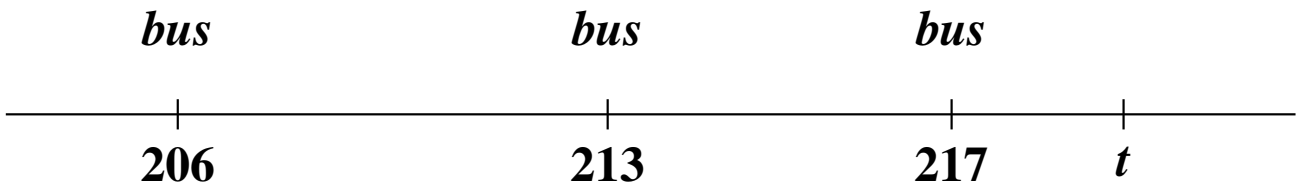
- "In the last hour, did a bus always appear in the last 5 minutes?"



- $\boxplus_i$: time-based window for last i minutes
- Query: $\boxplus_{60} \Box \boxplus_5$

Example: Nested Window

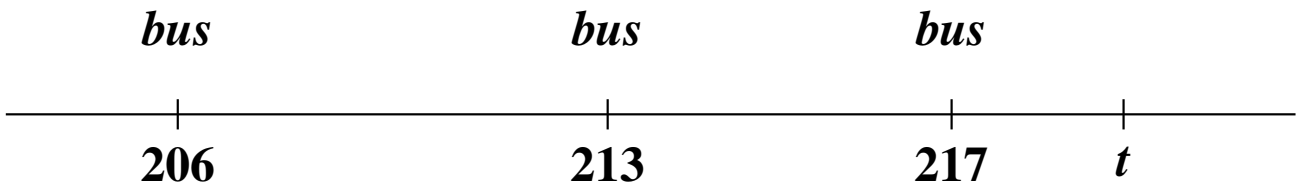
- "In the last hour, did a bus always appear in the last 5 minutes?"



- $\boxplus_i$: time-based window for last i minutes
- Query: $\boxplus_{60} \Box \boxplus_5 \Diamond$

Example: Nested Window

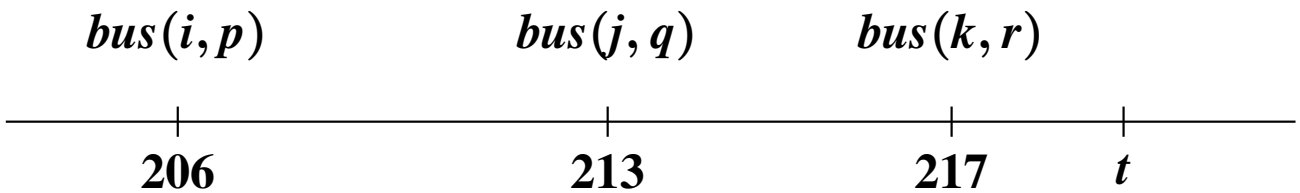
- "In the last hour, did a bus always appear in the last 5 minutes?"



- $\boxplus_i$: time-based window for last i minutes
- Query: $\boxplus_{60} \Box \boxplus_5 \Diamond bus$

Example: Nested Window

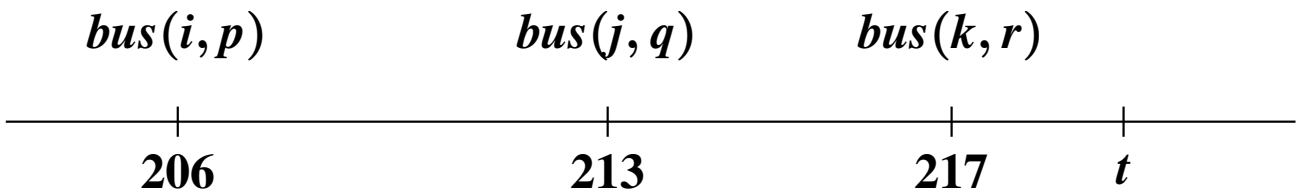
- "In the last hour, did a bus always appear in the last 5 minutes?"



- $\boxplus_i$: time-based window for last i minutes
- Query: $\boxplus_{60} \Box \boxplus_5 \Diamond bus$
- Limitation: $\boxplus_{60} \Box \boxplus_5 \Diamond bus(X, P)$

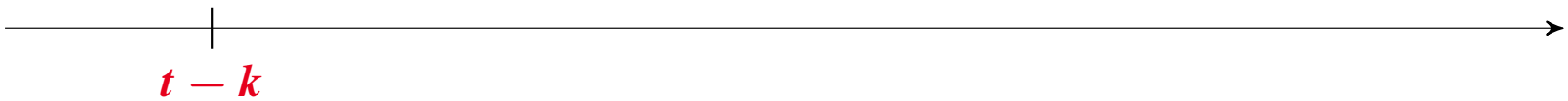
Example: Nested Window

- "In the last hour, did a bus always appear in the last 5 minutes?"



- $\boxplus_i$: time-based window for last i minutes
- Query: $\boxplus_{60} \Box \boxplus_5 \Diamond bus$
- Limitation: $\boxplus_{60} \Box \boxplus_5 \Diamond bus(X, P)$
  - Result: List of fixed combinations X, P
  - Need a rule: $some_bus \leftarrow bus(X, P)$
  - Then: $\boxplus_{60} \Box \boxplus_5 \Diamond some_bus$

Conclusion Stream

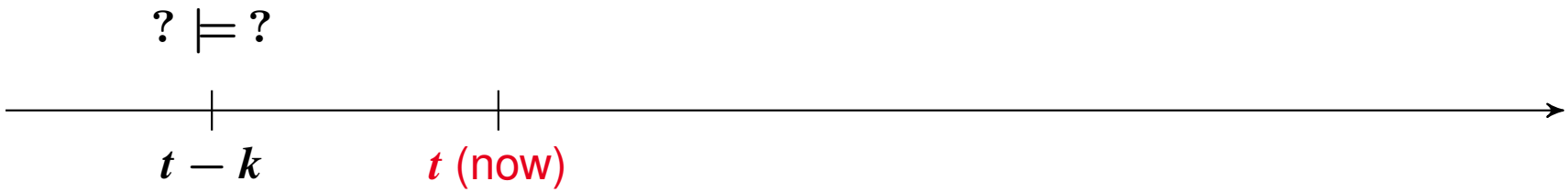


- Past

Conclusion Stream

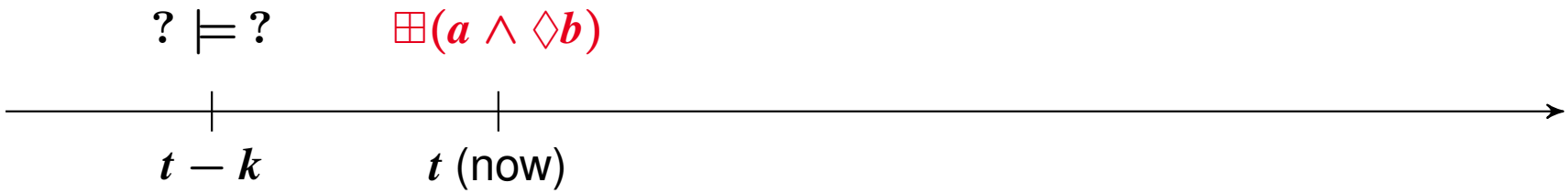
- Past: Lack of theoretical underpinning for stream reasoning

Conclusion Stream



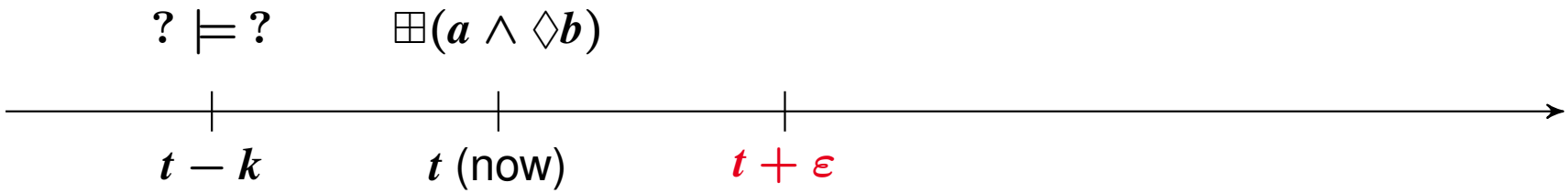
- Past: Lack of theoretical underpinning for stream reasoning
- Now

Conclusion Stream



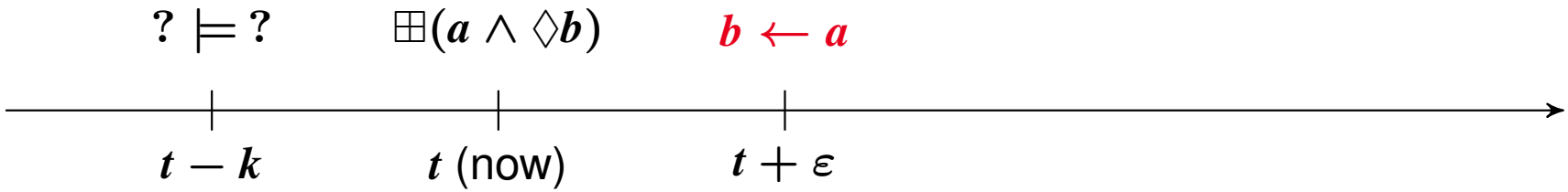
- Past: Lack of theoretical underpinning for stream reasoning
- Now: First language for modelling semantics precisely
  - flexible window operator (first class citizen)
  - time reference / time abstraction

Conclusion Stream



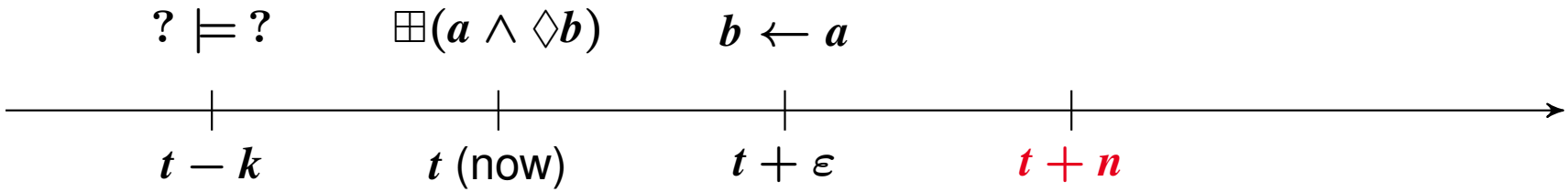
- Past: Lack of theoretical underpinning for stream reasoning
- Now: First language for modelling semantics precisely
  - flexible window operator (first class citizen)
  - time reference / time abstraction
- Soon

Conclusion Stream



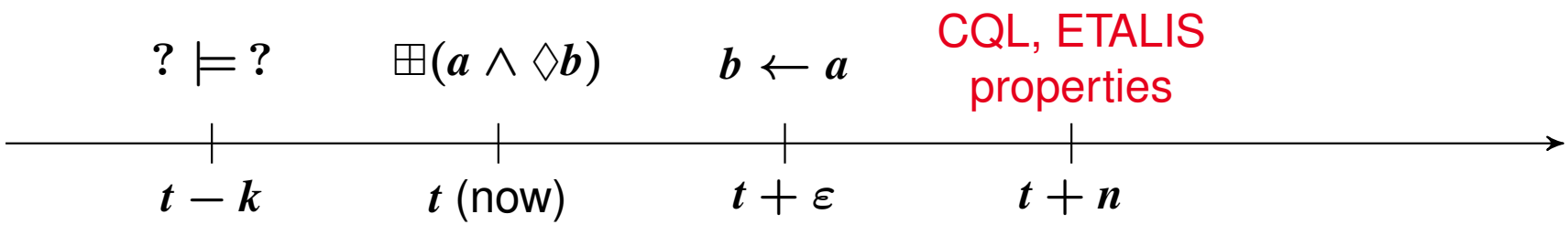
- Past: Lack of theoretical underpinning for stream reasoning
- Now: First language for modelling semantics precisely
  - flexible window operator (first class citizen)
  - time reference / time abstraction
- Soon: Rule-based extension (OrdRing @ ISWC, Oct.'14)

Conclusion Stream



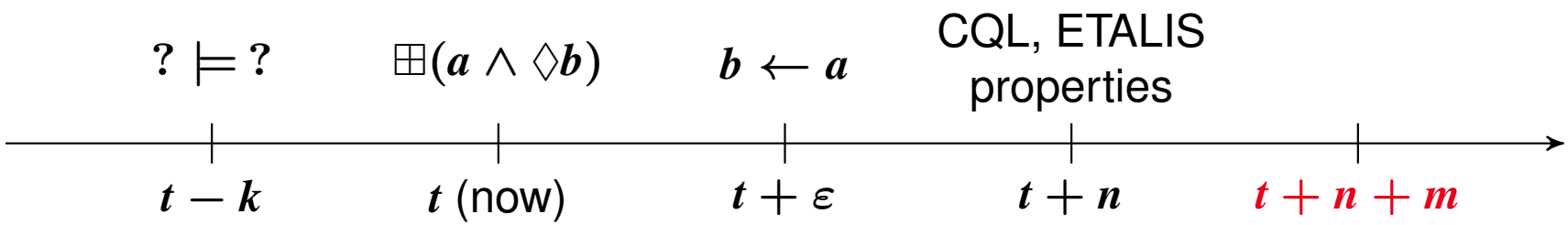
- Past: Lack of theoretical underpinning for stream reasoning
- Now: First language for modelling semantics precisely
  - flexible window operator (first class citizen)
  - time reference / time abstraction
- Soon: Rule-based extension (OrdRing @ ISWC, Oct.'14)
- Later

Conclusion Stream



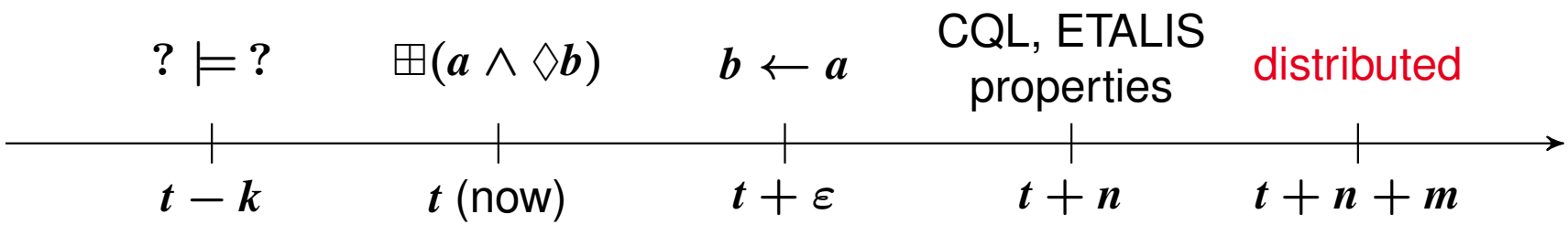
- Past: Lack of theoretical underpinning for stream reasoning
- Now: First language for modelling semantics precisely
  - flexible window operator (first class citizen)
  - time reference / time abstraction
- Soon: Rule-based extension (OrdRing @ ISWC, Oct.'14)
- Later: Language properties, capture CQL and ETALIS

Conclusion Stream



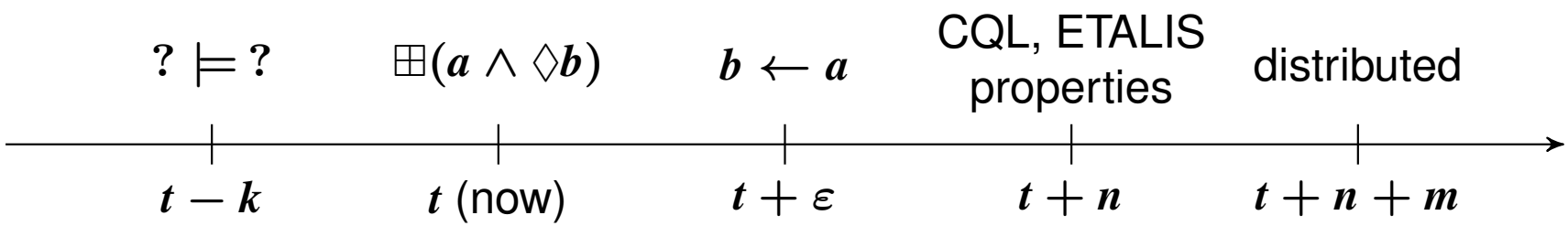
- Past: Lack of theoretical underpinning for stream reasoning
- Now: First language for modelling semantics precisely
  - flexible window operator (first class citizen)
  - time reference / time abstraction
- Soon: Rule-based extension (OrdRing @ ISWC, Oct.'14)
- Later: Language properties, capture CQL and ETALIS
- Eventually

Conclusion Stream



- Past: Lack of theoretical underpinning for stream reasoning
- Now: First language for modelling semantics precisely
  - flexible window operator (first class citizen)
  - time reference / time abstraction
- Soon: Rule-based extension (OrdRing @ ISWC, Oct.'14)
- Later: Language properties, capture CQL and ETALIS
- Eventually: Distributed setting, heterogeneous nodes

Conclusion Stream



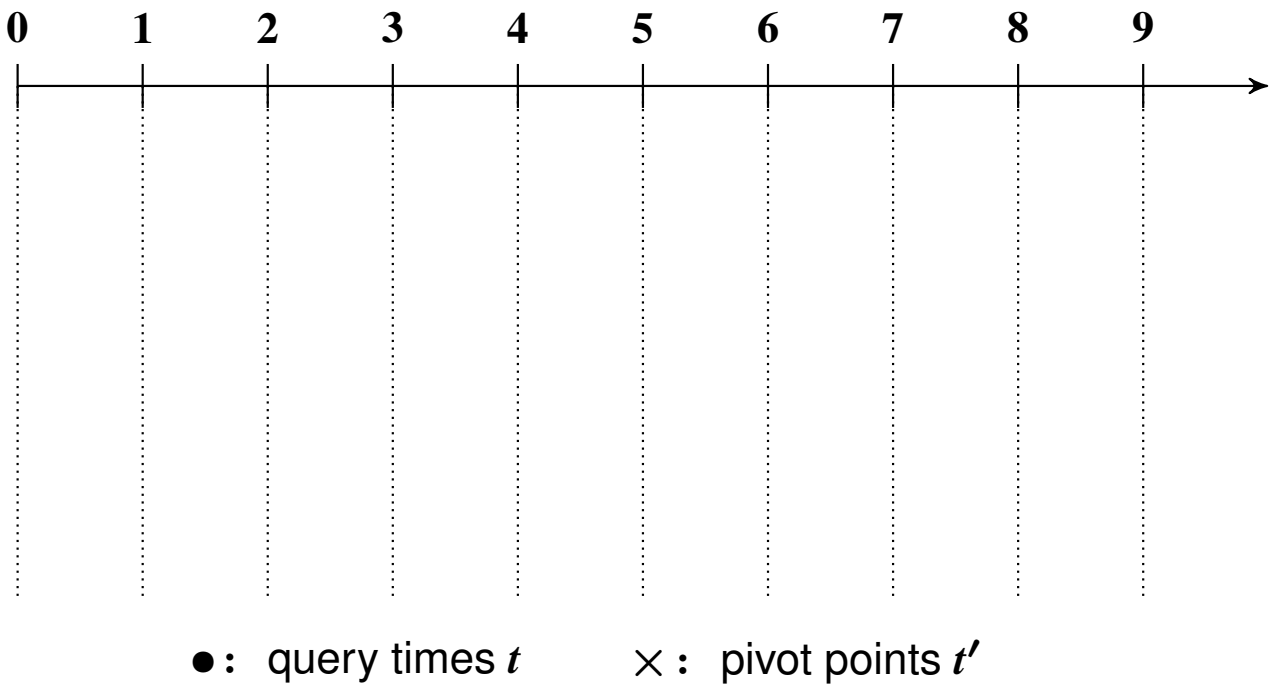
- Past: Lack of theoretical underpinning for stream reasoning
- Now: First language for modelling semantics precisely
  - flexible window operator (first class citizen)
  - time reference / time abstraction
- Soon: Rule-based extension (OrdRing @ ISWC, Oct.'14)
- Later: Language properties, capture CQL and ETALIS
- Eventually: Distributed setting, heterogeneous nodes

To je ono.

(That's it.)

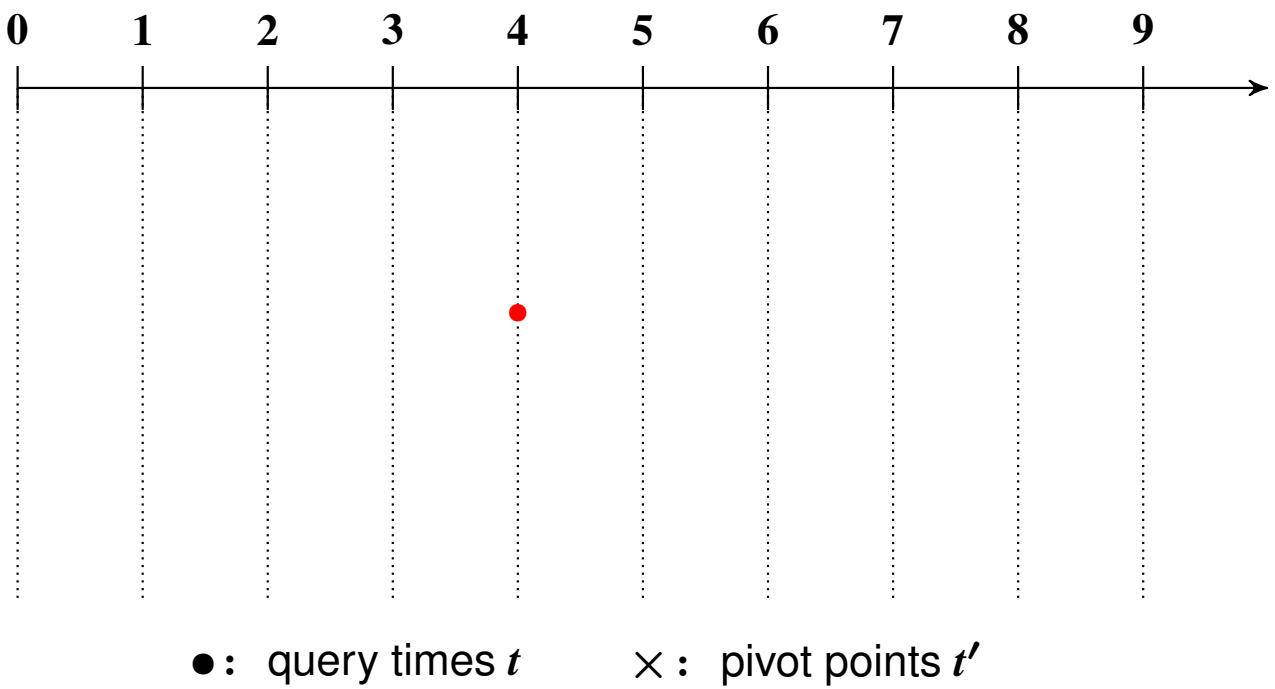
Time-based window

- Example
  - ℓ 2 time points into the past
  - u 1 time points into the future
  - d 3 step size (slide parameter)



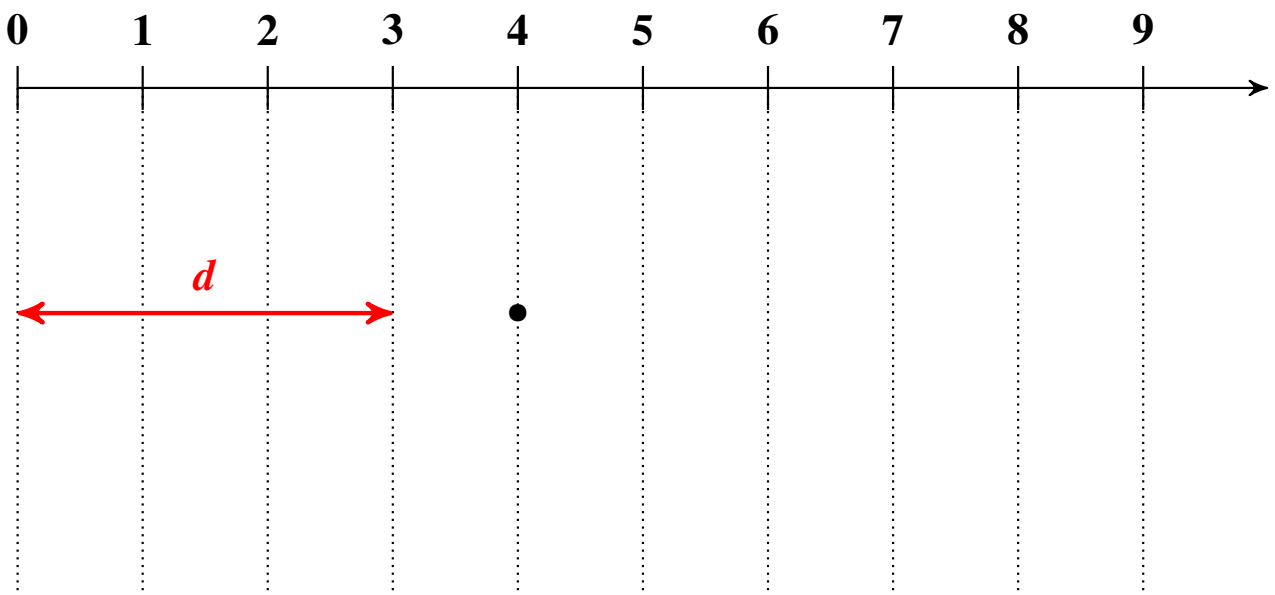
Time-based window

- Example: Query time $t = 4$
  - ℓ 2 time points into the past
  - u 1 time points into the future
  - d 3 step size (slide parameter)



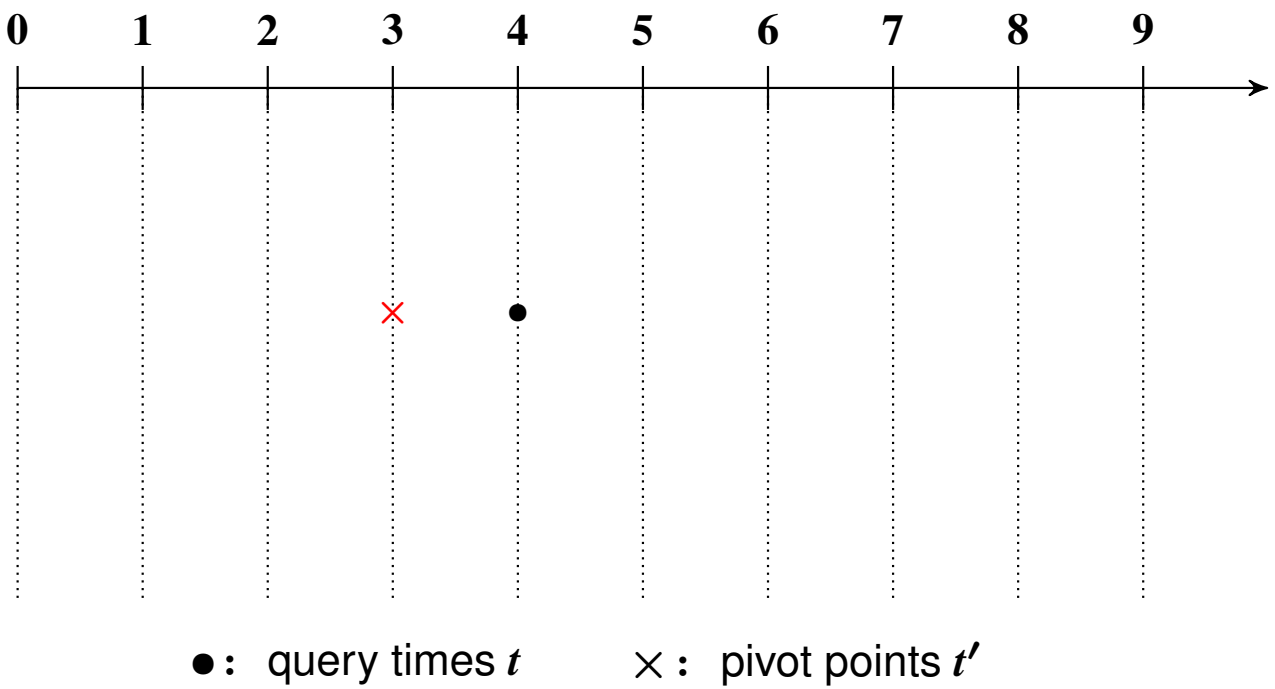
Time-based window

- Example: Query time $t = 4$
  - ℓ 2 time points into the past
  - u 1 time points into the future
  - d 3 step size (slide parameter)



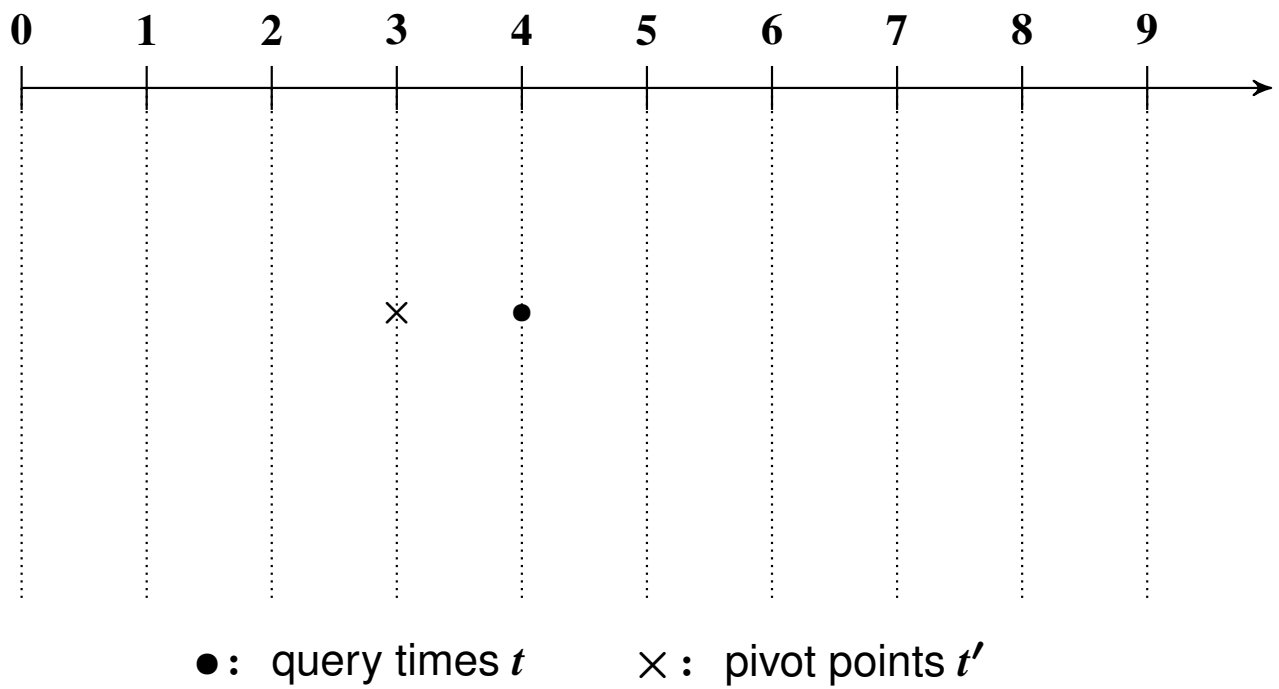
Time-based window

- Example: Query time $t = 4$
  - ℓ 2 time points into the past
  - u 1 time points into the future
  - d 3 step size (slide parameter)



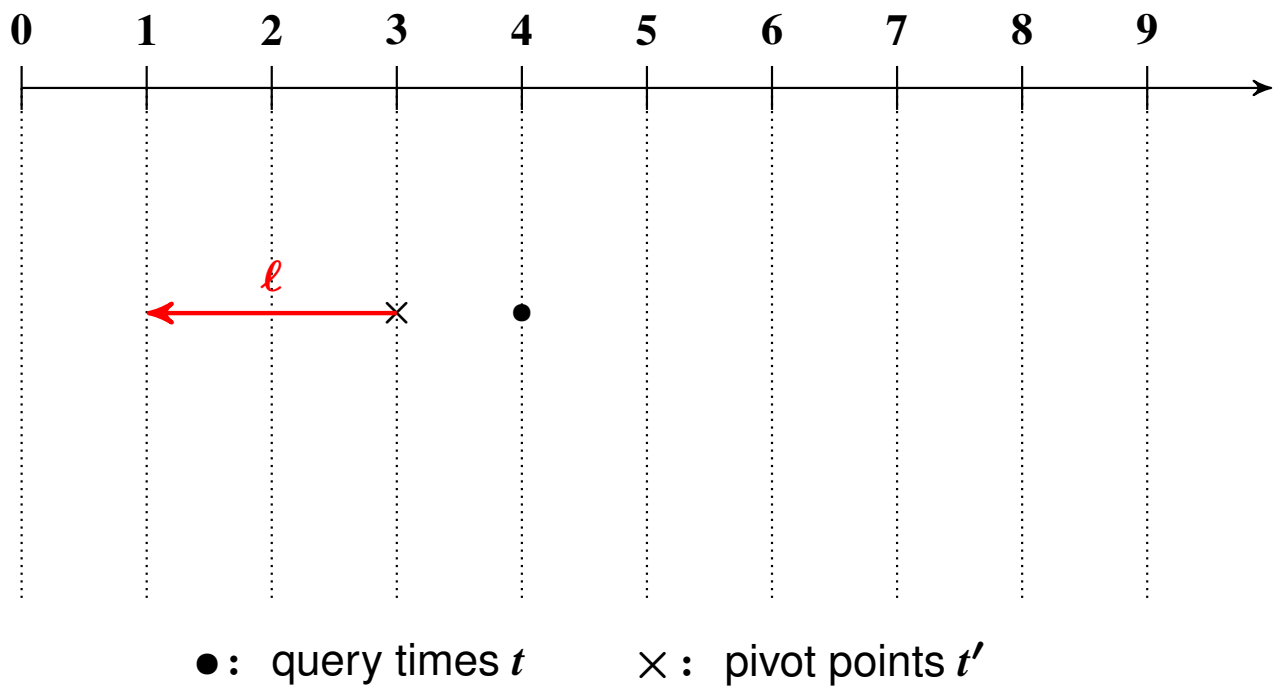
Time-based window

- Example: Query time $t = 4$
  - ℓ 2 time points into the past
  - u 1 time points into the future
  - d 3 step size (slide parameter)



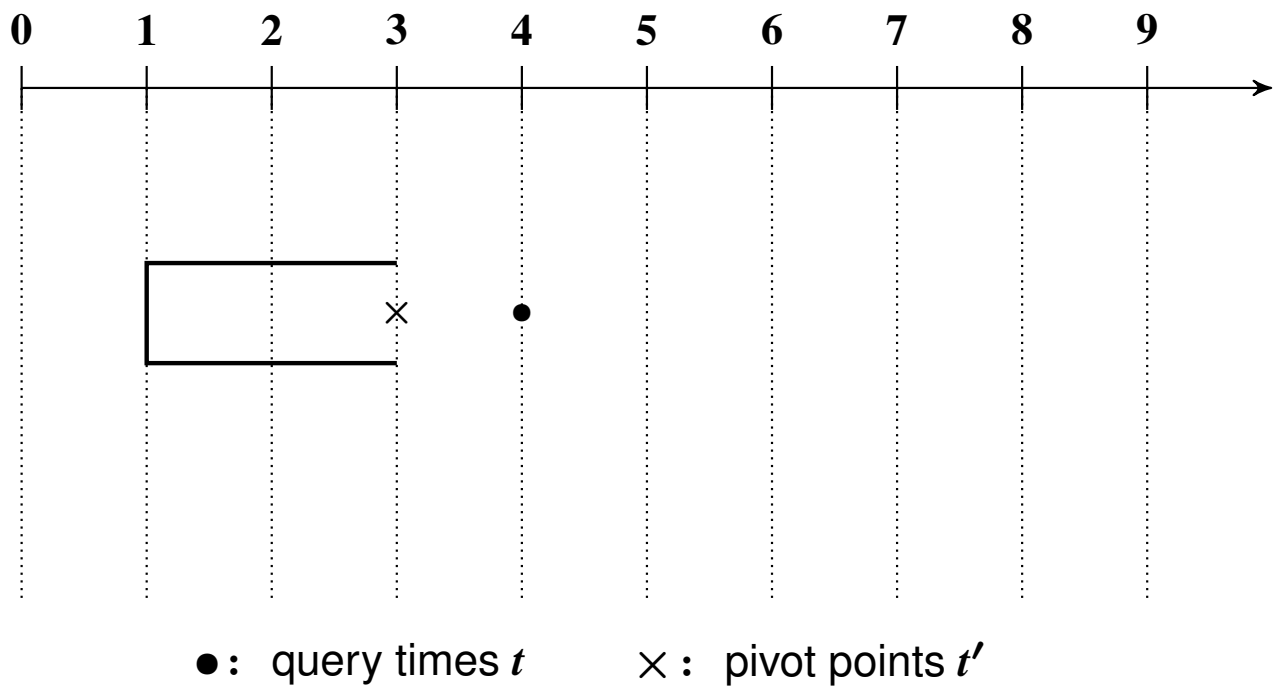
Time-based window

- Example: Query time $t = 4$
  - ℓ 2 time points into the past
  - u 1 time points into the future
  - d 3 step size (slide parameter)



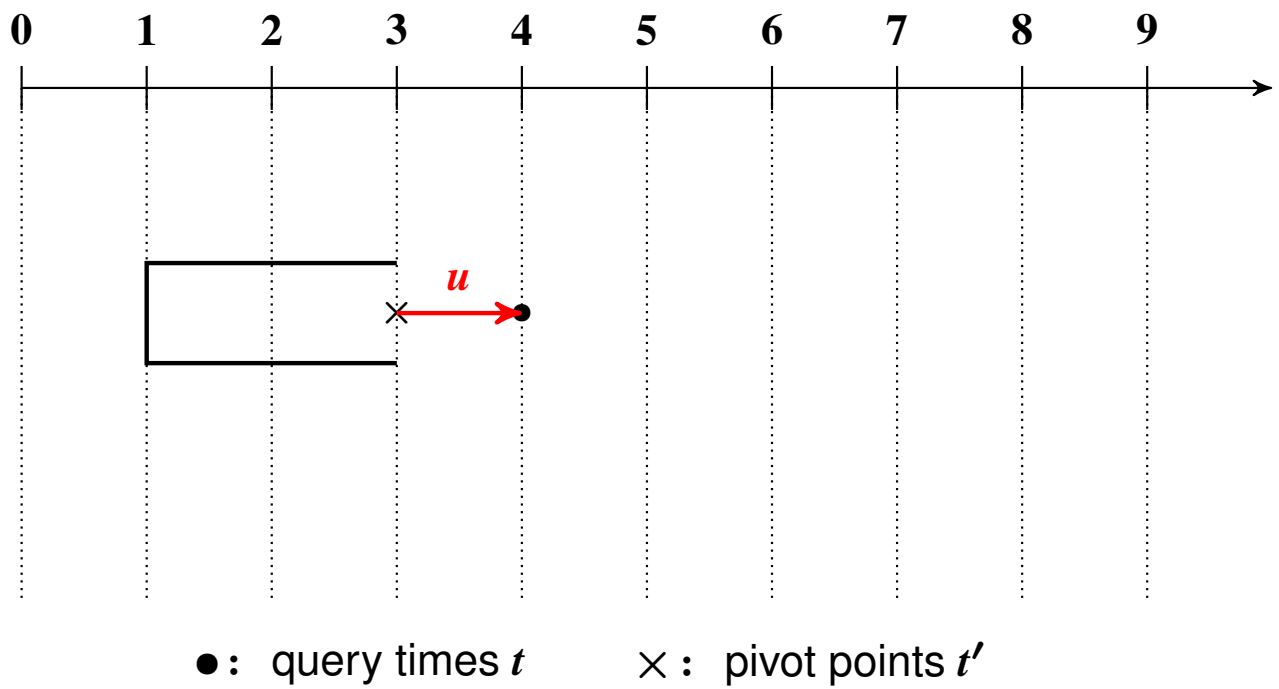
Time-based window

- Example: Query time $t = 4$
  - ℓ 2 time points into the past
  - u 1 time points into the future
  - d 3 step size (slide parameter)



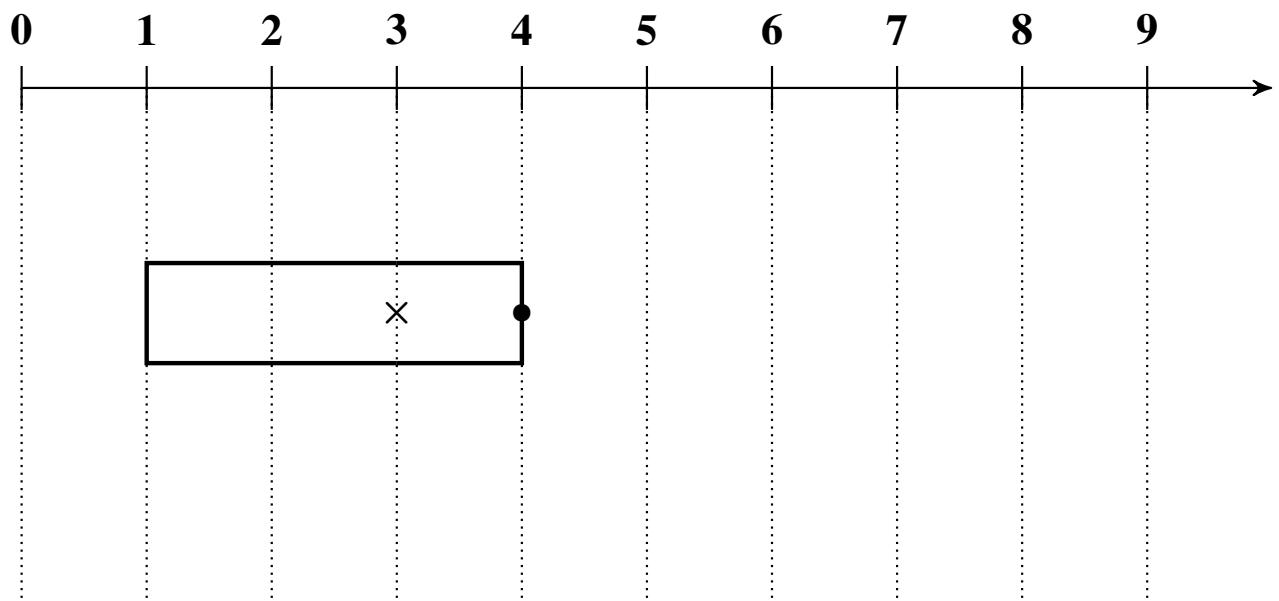
Time-based window

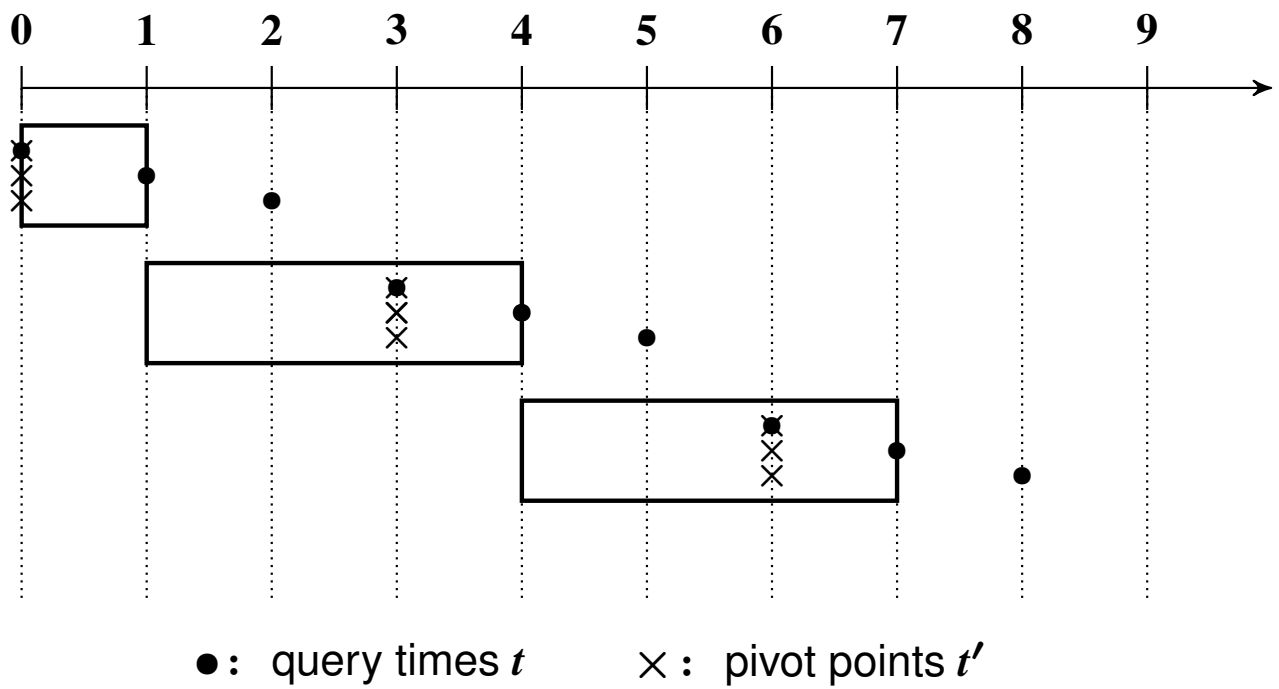
- Example: Query time $t = 4$
  - ℓ 2 time points into the past
  - u 1 time points into the future
  - d 3 step size (slide parameter)



Time-based window

- Example: Query time $t = 4$
  - ℓ 2 time points into the past
  - u 1 time points into the future
  - d 3 step size (slide parameter)



Time-based window

- Example
  - ℓ 2 time points into the past
  - u 1 time points into the future
  - d 3 step size (slide parameter)