Raising the Level of Abstraction for Time State Analytics With the Timeline Framework

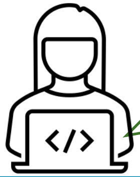
Henry Milner, Yihua Cheng, Jibin Zhan,

Hui Zhang, Vyas Sekar, Junchen Jiang, Ion Stoica

Conviva, Carnegie Mellon, UChicago, UC Berkeley

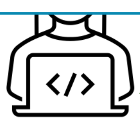
10 January 2023

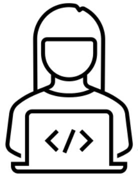
Example Analytics Query Intents

Video  How much time did this session spend buffering while using CDN C1? → Change CDN

Common feature:

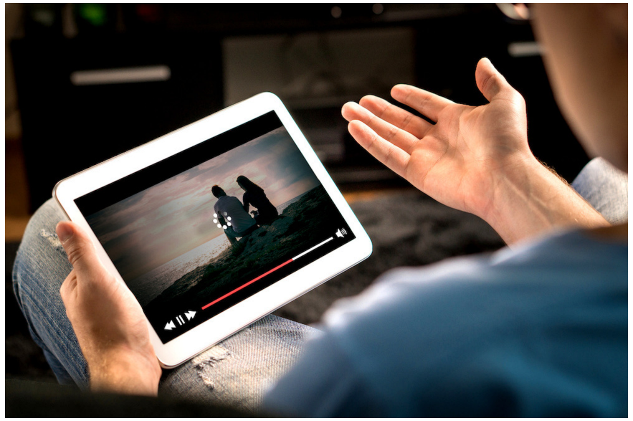
Stateful context-sensitive metrics computed over continuous time

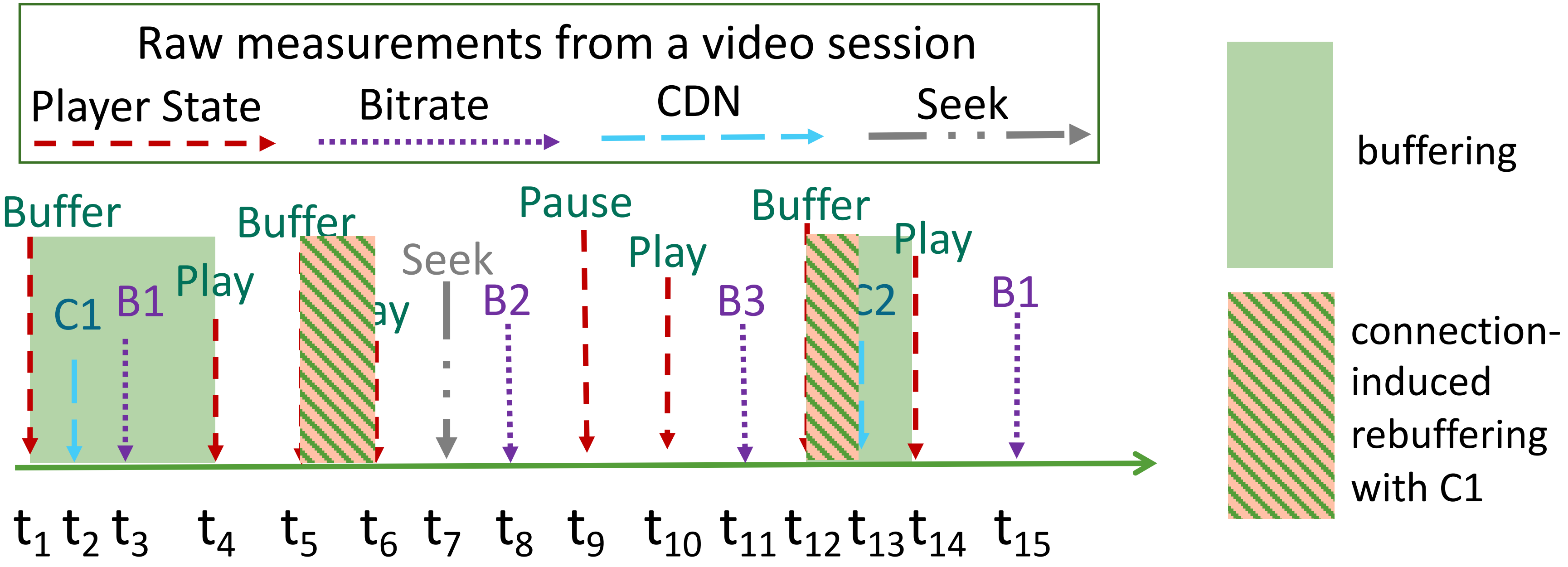
Cybersecurity  visiting website xyz.com in the last hour? → Quarantine host

Manufacturing  How many machines from vendor X are showing degrading health status over time? → Rebalance load, Repair

…

Streaming Video QoE: Connection-Induced Rebuffering





Stateful

Context-sensitive

Continuous time

How much time did this session spend in a connection-induced rebuffering state while using CDN C1?

Count the duration where:

1. Currently buffering &
2. Play has already initialized &
3. Hasn’t seeked in last 5 seconds &
4. Using CDN C1

Time-State Analytics (TSA), in a nutshell

Video

How much time did this session spend buffering while using CDN C1?

Change CDN

Finance

raud alert, block

Compute ***stateful, context-sensitive*** metrics over ***continuous time,*** using ***measurements at scale.***

Cybersecurity

visiting website xyz.com in the last hour in ATL?

lock URL
Quarantine host

Manufacturing

How many machines from vendor X are showing poor health status in Florida ?

Rebalance load, Reboot

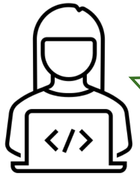
…

TSA isn’t served well by existing data processing systems

```
WITH SeekAsPlayerState(T, P) as (
  SELECT T, P FROM heartbeats WHERE P IS NOT NULL
  UNION SELECT T, "Seek_st" FROM heartbeats WHERE A IS NOT NULL
  UNION SELECT T + 5, "Seek_ed" FROM heartbeats WHERE A IS NOT NULL ),
IgnoreBufBeforePlay(T, P) as (
  SELECT T, P FROM (
    SELECT T, P, Max(If(P == 'play', 1, 0)) OVER (PARTITION BY 1 ORDER BY T)
      ↪ as H
    FROM SeekAsPlayerState) WHERE H == True ),
DuringBufferTable(T, P, DB) as (
  SELECT T, P, LAST(tmp1) IGNORE NULLS OVER (PARTITION BY 1 ORDER BY T)
  FROM (
    SELECT T, P,
          CASE P WHEN 'buffer' THEN True WHEN 'Seek_st' THEN NULL WHEN 'Seek_ed'
            ↪ THEN NULL ELSE FALSE END as tmp1
    From IgnoreBufBeforePlay ) ),
DuringSeekTable(T, P, DB, DS) as (
  SELECT T, P, DB,
          (T - Max(If(P == 'Seek_st', T, 0)) OVER (PARTITION BY 1 ORDER BY T)
            ↪ ) < 5 as tmp2
  FROM DuringBufferTable ),
IgnoreBufInSeek(T, P) as (
  SELECT T, P FROM (
    SELECT T, DS, IF(P == 'Seek_ed' and DB, 'buffer', P) as P
   FROM DuringSeekTable ) WHERE NOT (P == 'buffer' AND DS) ),
WithCDNAndQuery(T, P, C) as (
 SELECT T, P, NULL FROM IgnoreBufInSeek
 UNION SELECT T, NULL, C FROM heartbeats where C IS NOT NULL
 UNION SELECT 2022-07-21 10:05, NULL, NULL s),
Intervals(Ed, St, State, CDN) as (
  SELECT T, LEAD(T, 1) OVER (PARTITION BY 1 ORDER BY T), P, C
  FROM (
    SELECT T,
          LAST(P) IGNORE NULLS OVER (PARTITION BY 1 ORDER BY T) as P,
          LAST(C) IGNORE NULLS OVER (PARTITION BY 1 ORDER BY T) as C
    FROM WithCDNAndQuery ) )
SELECT SUM(St - Ed) as result FROM Intervals
WHERE Ed < 2022-07-21 10:05 AND State == 'buffer' AND CDN == 'CDN1'
```

High dev effort

High cost



Count the duration where:

1. Currently buffering &
2. Play has already initialized &
3. Hasn’t seeked in last 5 seconds &
4. Using CDN C1

Our work: Timeline abstraction for Time-State Analytics

Writing time-state queries becomes intuitive visual operations
→ Reduced dev effort

Enables new opportunities for structure-aware optimizations
→ Up to 10x improvement in cost

Scope of this work: Focus on the single user-session intent modeling problem
Outside our scope: Supporting scale-out and aggregation

Outline for talk

- What is Time-State Analytics
- ***Time-State Analytics not well supported by status quo***
- Introducing the Timeline abstraction
- Early Wins + Next Steps

Where’s the problem?

```
WITH SeekAsPlayerState(T, P) as (
  SELECT T, P FROM heartbeats WHERE P IS NOT NULL
  UNION SELECT T, "Seek_st" FROM heartbeats WHERE A IS NOT NULL
  UNION SELECT T + 5, "Seek_ed" FROM heartbeats WHERE A IS NOT NULL ),
IgnoreBufBeforePlay(T, P) as (
  SELECT T, P FROM (
    SELECT T, P, Max(If(P == 'play', 1, 0)) OVER (PARTITION BY 1 ORDER BY T)
      ↪ as H
    FROM SeekAsPlayerState) WHERE H == True ),
DuringBufferTable(T, P, DB) as (
  SELECT T, P, LAST(tmp1) IGNORE NULLS OVER (PARTITION BY 1 ORDER BY T)
  FROM (
    SELECT T, P,
          CASE P WHEN 'buffer' THEN True WHEN 'Seek_st' THEN NULL WHEN 'Seek_ed'
            ↪ THEN NULL ELSE FALSE END as tmp1
    From IgnoreBufBeforePlay ) ),
DuringSeekTable(T, P, DB, DS) as (
  SELECT T, P, DB,
          (T - Max(If(P == 'Seek_st', T, 0)) OVER (PARTITION BY 1 ORDER BY T)
           ↪ ) < 5 as tmp2
  FROM DuringBufferTable ),
IgnoreBufInSeek(T, P) as (
  SELECT T, P FROM (
    SELECT T, DS, IF(P == 'Seek_ed' and DB, 'buffer', P) as P
   FROM DuringSeekTable ) WHERE NOT (P == 'buffer' AND DS) ),
WithCDNAndQuery(T, P, C) as (
 SELECT T, P, NULL FROM IgnoreBufInSeek
 UNION SELECT T, NULL, C FROM heartbeats where C IS NOT NULL
 UNION SELECT 2022-07-21 10:05, NULL, NULL s),
Intervals(Ed, St, State, CDN) as (
  SELECT T, LEAD(T, 1) OVER (PARTITION BY 1 ORDER BY T), P, C
  FROM (
    SELECT T,
          LAST(P) IGNORE NULLS OVER (PARTITION BY 1 ORDER BY T) as P,
          LAST(C) IGNORE NULLS OVER (PARTITION BY 1 ORDER BY T) as C
    FROM WithCDNAndQuery ) )
SELECT SUM(St - Ed) as result FROM Intervals
WHERE Ed < 2022-07-21 10:05 AND State == 'buffer' AND CDN == 'CDN1'
```

Google BigQuery

databricks Lakehouse Platform

snowflake

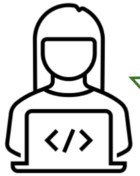
Google Dataflow

Flink

Spark Streaming

High dev effort

High cost

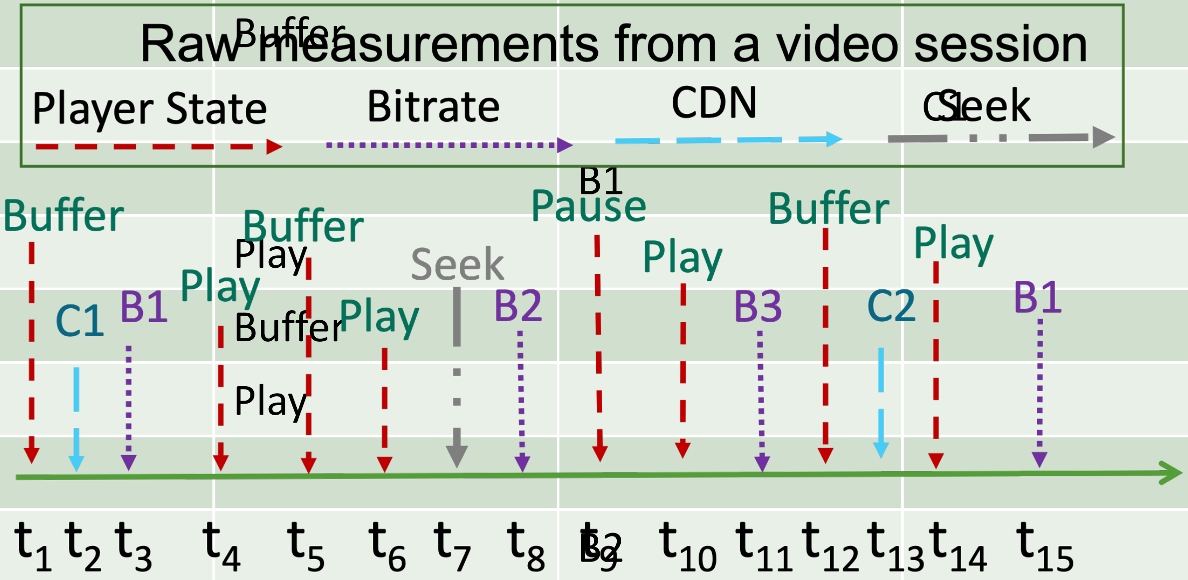


Count the duration where:

1. Currently buffering &
2. Play has already initialized &
3. Hasn’t seeked in last 5 seconds &
4. Using CDN C1

Tabular model isn't well-suited for Time-State

Timestamp	Player State	Bitrate	CDN	Seek
t1	Buffer			
t2			C1	
t3		B1		
t4	Play			
t5	Buffer			
t6	Play			
t7				Seek
t8		B2		
t9	Paused			
t10	Play			
t11		B3		
t12	Buffer			
t13			C2	
t14	Play			
t15		B1		



Count the duration where:

1. Currently buffering &
2. Play has already initialized &
3. Hasn't seeked in last 5 seconds &
4. Using CDN C1

State and Context over Continuous Time is Hard

Timestamp	Player State	Bitrate	CDN	Seek
t1	Buffer			
t2	Buffer		C1	
t3	Buffer			
t4	Play			
t5	Buffer			
t6	Play			
t7	Play			
t8	Play			
t9	Paused			
t10	Play			
t11	Play			
t12	Buffer			
t13	Buffer	B3	C2	
t14	Play	B3	C2	
t15	Play	B1	C2	

duration?

```
[…]

WITH SeekAsPlayerState(T, P) as (
  SELECT T, P FROM heartbeats WHERE P IS NOT NULL
  UNION SELECT T, "Seek_st" FROM heartbeats WHERE A IS NOT NULL
  UNION SELECT T + 5, "Seek_ed" FROM heartbeats WHERE A IS NOT NULL ),
[…]
DuringBufferTable(T, P, DB) as (
  SELECT T, P, LAST(tmp1) IGNORE NULLS OVER (PARTITION BY 1 ORDER BY T)
  FROM (
    SELECT T, P,
          CASE P WHEN 'buffer' THEN True WHEN 'Seek_st' THEN NULL WHEN 'Seek_ed'
            ↪ THEN NULL ELSE FALSE END as tmp1
    From IgnoreBufBeforePlay ) ),
DuringSeekTable(T, P, DB, DS) as (
  SELECT T, P, DB,
          (T - Max(If(P == 'Seek_st', T, 0)) OVER (PARTITION BY 1 ORDER BY T)
            ↪ ) < 5 as tmp2
  FROM DuringBufferTable ),
IgnoreBufInSeek(T, P) as (
  SELECT T, P FROM (
    SELECT T, DS, IF(P == 'Seek_ed' and DB, 'buffer', P) as P
   FROM DuringSeekTable ) WHERE NOT (P == 'buffer' AND DS) ),

[…]
```

Count the duration where:

1. Currently buffering &
2. Play has already initialized &
3. Hasn't seeked in last 5 seconds &
4. Using CDN C1

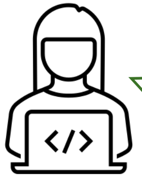
t7 + 5 seconds???

Poor abstraction → Complex code

```sql
WITH SeekAsPlayerState(T, P) as (
  SELECT T, P FROM heartbeats WHERE P IS NOT NULL
  UNION SELECT T, "Seek_st" FROM heartbeats WHERE A IS NOT NULL
  UNION SELECT T + 5, "Seek_ed" FROM heartbeats WHERE A IS NOT NULL ),
IgnoreBufBeforePlay(T, P) as (
  SELECT T, P FROM (
    SELECT T, P, Max(If(P == 'play', 1, 0)) OVER (PARTITION BY 1 ORDER BY T)
      ↪ as H
    FROM SeekAsPlayerState) WHERE H == True ),
DuringBufferTable(T, P, DB) as (
  SELECT T, P, LAST(tmp1) IGNORE NULLS OVER (PARTITION BY 1 ORDER BY T)
  FROM (
    SELECT T, P,
          CASE P WHEN 'buffer' THEN True WHEN 'Seek_st' THEN NULL WHEN 'Seek_ed'
            ↪ THEN NULL ELSE FALSE END as tmp1
    From IgnoreBufBeforePlay ) ),
DuringSeekTable(T, P, DB, DS) as (
  SELECT T, P, DB,
          (T - Max(If(P == 'Seek_st', T, 0)) OVER (PARTITION BY 1 ORDER BY T)
            ↪ ) < 5 as tmp2
  FROM DuringBufferTable ),
IgnoreBufInSeek(T, P) as (
  SELECT T, P FROM (
    SELECT T, DS, IF(P == 'Seek_ed' and DB, 'buffer', P) as P
   FROM DuringSeekTable ) WHERE NOT (P == 'buffer' AND DS) ),
WithCDNAndQuery(T, P, C) as (
 SELECT T, P, NULL FROM IgnoreBufInSeek
 UNION SELECT T, NULL, C FROM heartbeats where C IS NOT NULL
 UNION SELECT 2022-07-21 10:05, NULL, NULL s),
Intervals(Ed, St, State, CDN) as (
  SELECT T, LEAD(T, 1) OVER (PARTITION BY 1 ORDER BY T), P, C
  FROM (
    SELECT T,
          LAST(P) IGNORE NULLS OVER (PARTITION BY 1 ORDER BY T) as P,
          LAST(C) IGNORE NULLS OVER (PARTITION BY 1 ORDER BY T) as C
    FROM WithCDNAndQuery ) )
SELECT SUM(St - Ed) as result FROM Intervals
WHERE Ed < 2022-07-21 10:05 AND State == 'buffer' AND CDN == 'CDN1'
```

Difficult to develop

Semantic bugs



Count the duration where:
1. Currently buffering &
2. Play has already initialized &
3. Hasn't seeked in last 5 seconds &
4. Using CDN C1

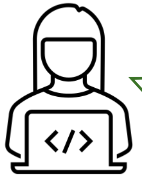
Poor abstraction → High cost

```
WITH SeekAsPlayerState(T, P) as (
  SELECT T, P FROM heartbeats WHERE P IS NOT NULL
  UNION SELECT T, "Seek_st" FROM heartbeats WHERE A IS NOT NULL
  UNION SELECT T + 5, "Seek_ed" FROM heartbeats WHERE A IS NOT NULL ),
IgnoreBufBeforePlay(T, P) as (
  SELECT T, P FROM (
    SELECT T, P, Max(If(P == 'play', 1, 0)) OVER (PARTITION BY 1 ORDER BY T) as H
    FROM SeekAsPlayerState) WHERE H == True ),
DuringBufferTable(T, P, DB) as (
  SELECT T, P, LAST(tmp1) IGNORE NULLS OVER (PARTITION BY 1 ORDER BY T)
  FROM (
    SELECT T, P,
          CASE P WHEN 'buffer' THEN True WHEN 'Seek_st' THEN NULL WHEN 'Seek_ed' THEN NULL ELSE FALSE END as tmp1
    From IgnoreBufBeforePlay ) ),
DuringSeekTable(T, P, DB, DS) as (
  SELECT T, P, DB,
          (T - Max(If(P == 'Seek_st', T, 0)) OVER (PARTITION BY 1 ORDER BY T) ) < 5 as tmp2
  FROM DuringBufferTable ),
IgnoreBufInSeek(T, P) as (
  SELECT T, P FROM (
    SELECT T, DS, IF(P == 'Seek_ed' and DB, 'buffer', P) as P
   FROM DuringSeekTable ) WHERE NOT (P == 'buffer' AND DS) ),
WithCDNAndQuery(T, P, C) as (
 SELECT T, P, NULL FROM IgnoreBufInSeek
 UNION SELECT T, NULL, C FROM heartbeats where C IS NOT NULL
 UNION SELECT 2022-07-21 10:05, NULL, NULL s),
Intervals(Ed, St, State, CDN) as (
  SELECT T, LEAD(T, 1) OVER (PARTITION BY 1 ORDER BY T), P, C
  FROM (
    SELECT T,
          LAST(P) IGNORE NULLS OVER (PARTITION BY 1 ORDER BY T) as P,
          LAST(C) IGNORE NULLS OVER (PARTITION BY 1 ORDER BY T) as C
    FROM WithCDNAndQuery ) )
SELECT SUM(St - Ed) as result FROM Intervals
WHERE Ed < 2022-07-21 10:05 AND State == 'buffer' AND CDN == 'CDN1'
```

Lacks structure:

Difficult for query engines to optimize

High cost



Count the duration where:

1. Currently buffering &
2. Play has already initialized &
3. Hasn’t seeked in last 5 seconds &
4. Using CDN C1

Outline for talk

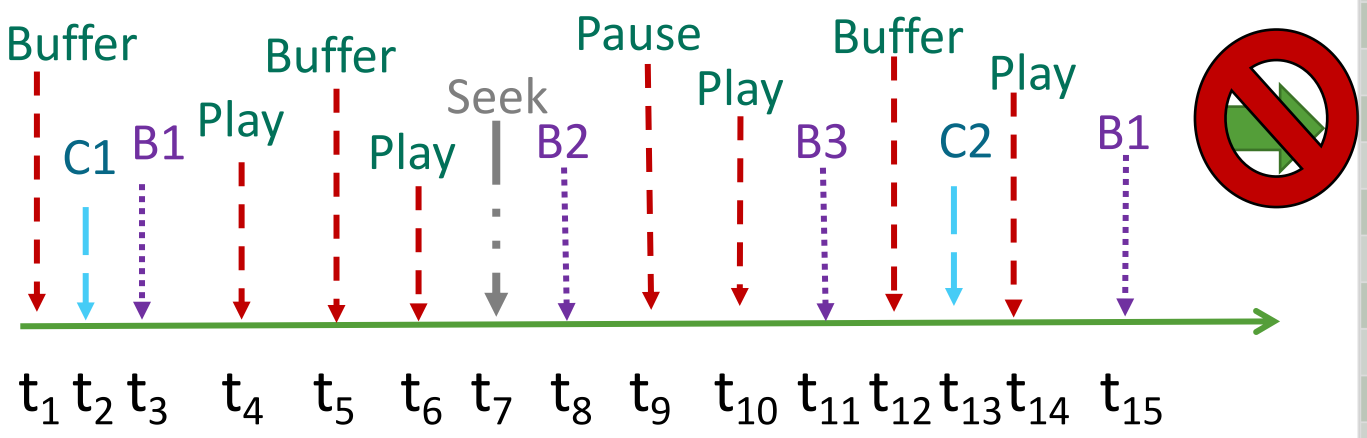
- What is Time-State Analytics

- Time-State Analytics not well supported by status quo

- ***Introducing the Timeline abstraction***

- Early Wins + Next Steps

Stepping back



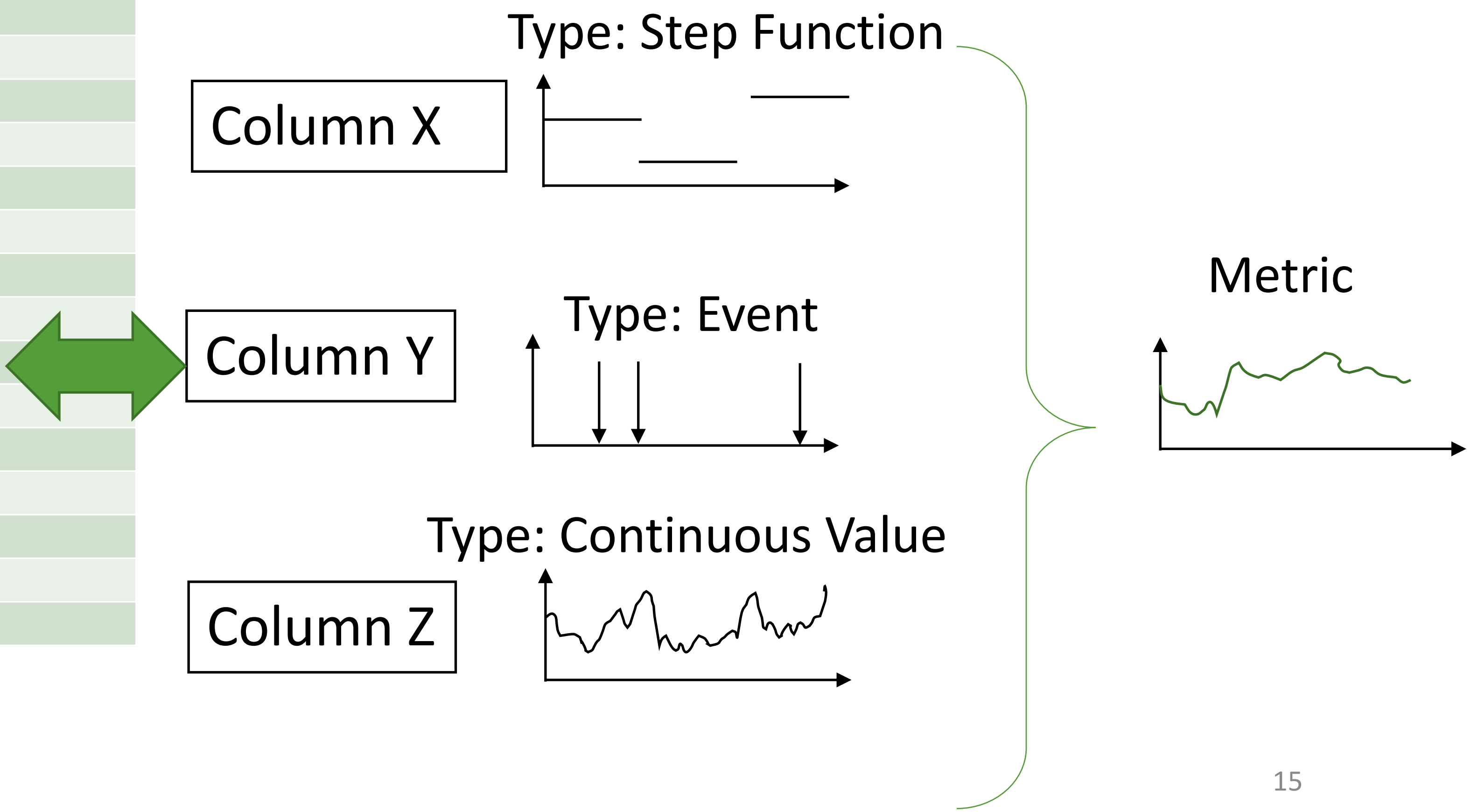
Timestamp	Player State	Bitrate	CDN	Seek
t1	Buffer			
t2			C1	
t3		B1		
t4	Play			
t5	Buffer			
t6	Play			
t7				Seek
t8		B2		
t9	Paused			
t10	Play			
t11		B3		
t12	Buffer			
t13			C2	
t14	Play			
t15		B1		

What’s a Timeline?

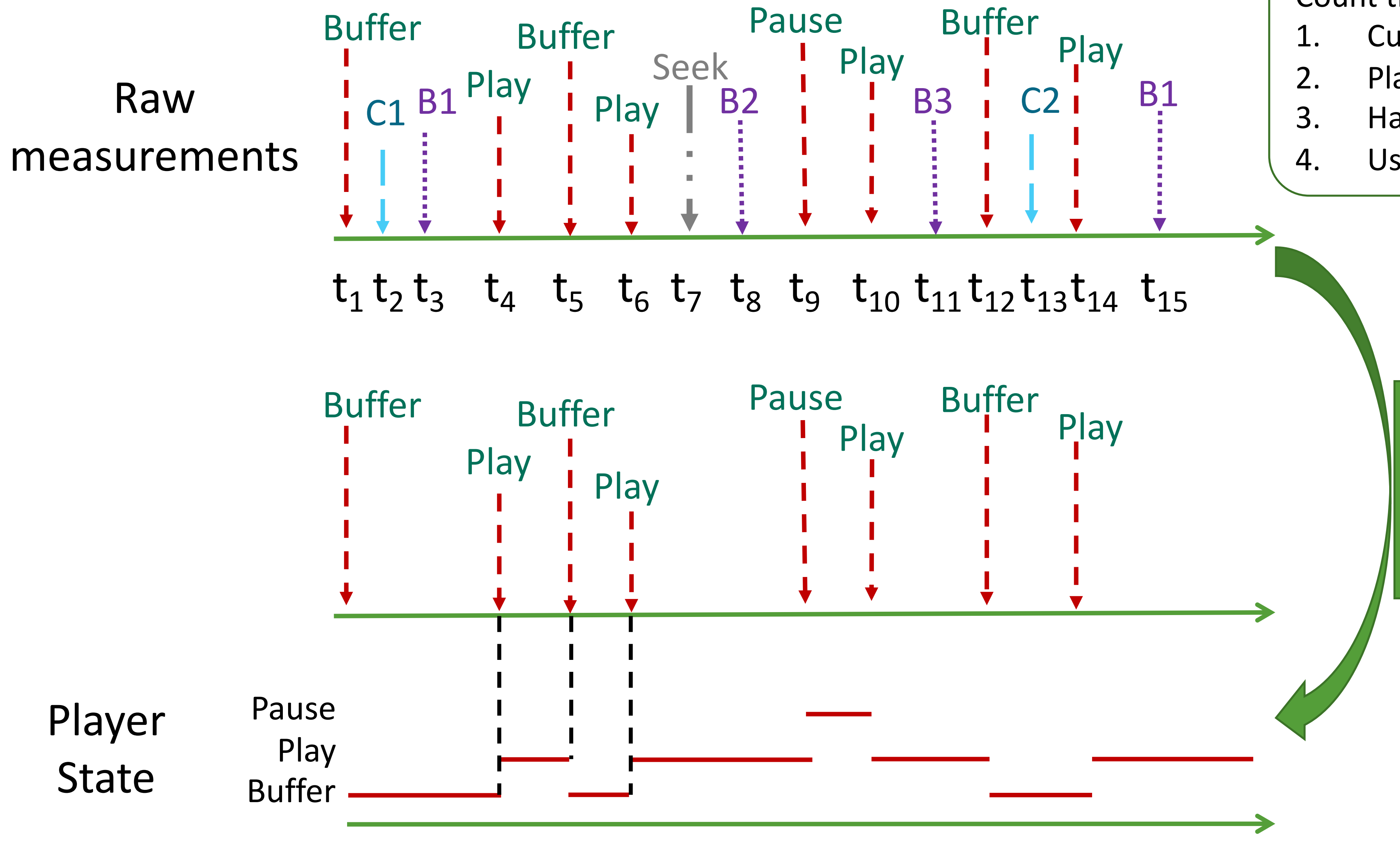
“Geometric abstractions are powerful tools” *– Fred Brooks*

An equivalent *geometric* view:
Timeline of each measurement

Timestamp	Player State	Bitrate	CDN	Seek
t1	Buffer			
t2			C1	
t3		B1		
t4	Play			
t5	Buffer			
t6	Play			
t7				Seek
t8		B2		
t9	Paused			
t10	Play			
t11		B3		
t12	Buffer			
t13			C2	
t14	Play			
t15		B1		



Whiteboarding Timelines: the Player State over Time



Count the duration where:

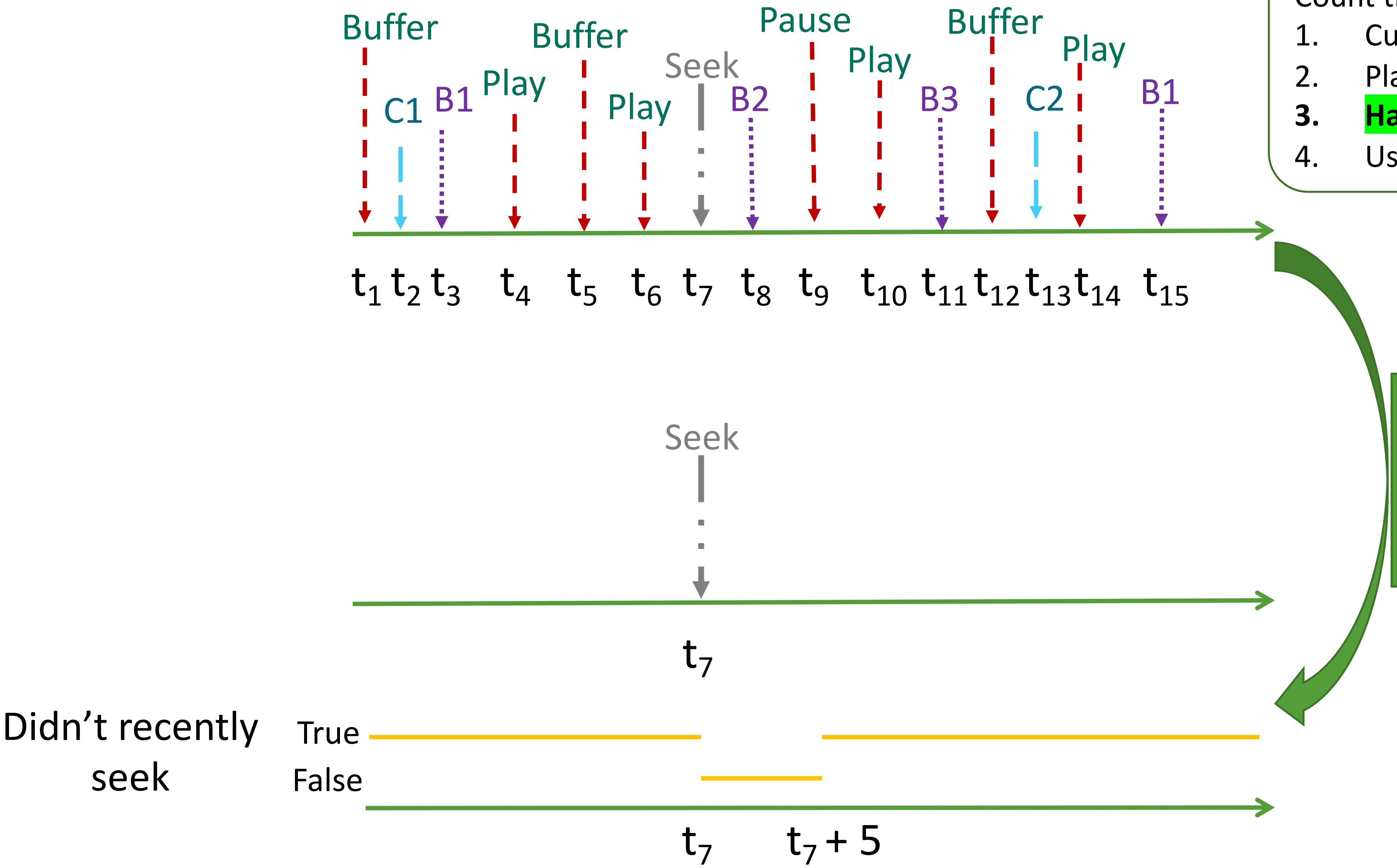
1. Currently buffering &
2. Play has already initialized &
3. Hasn’t seeked in last 5 seconds &
4. Using CDN C1

Operator: LatestEventToState (PlayerState)

Whiteboarding Timelines: When has a Seek recently happened?

Count the duration where:

1. Currently buffering &
2. Play has already initialized &
3. **Hasn't seeked in last 5 seconds** &
4. Using CDN C1

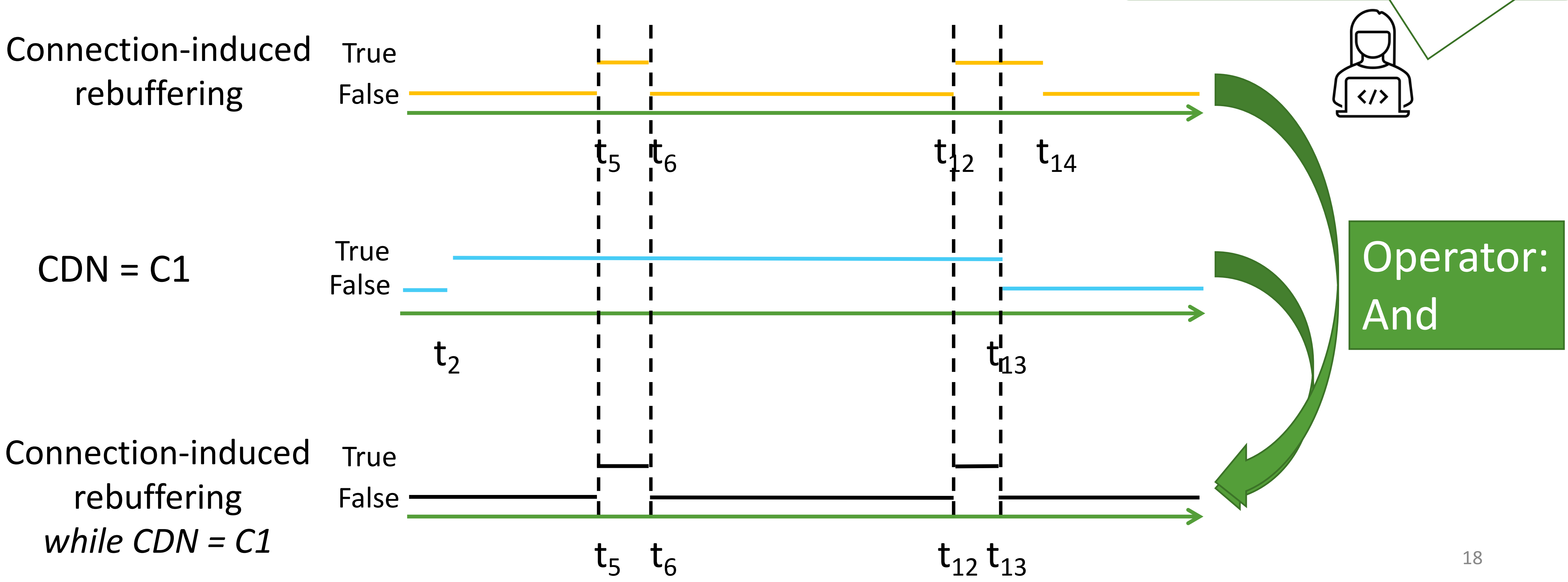


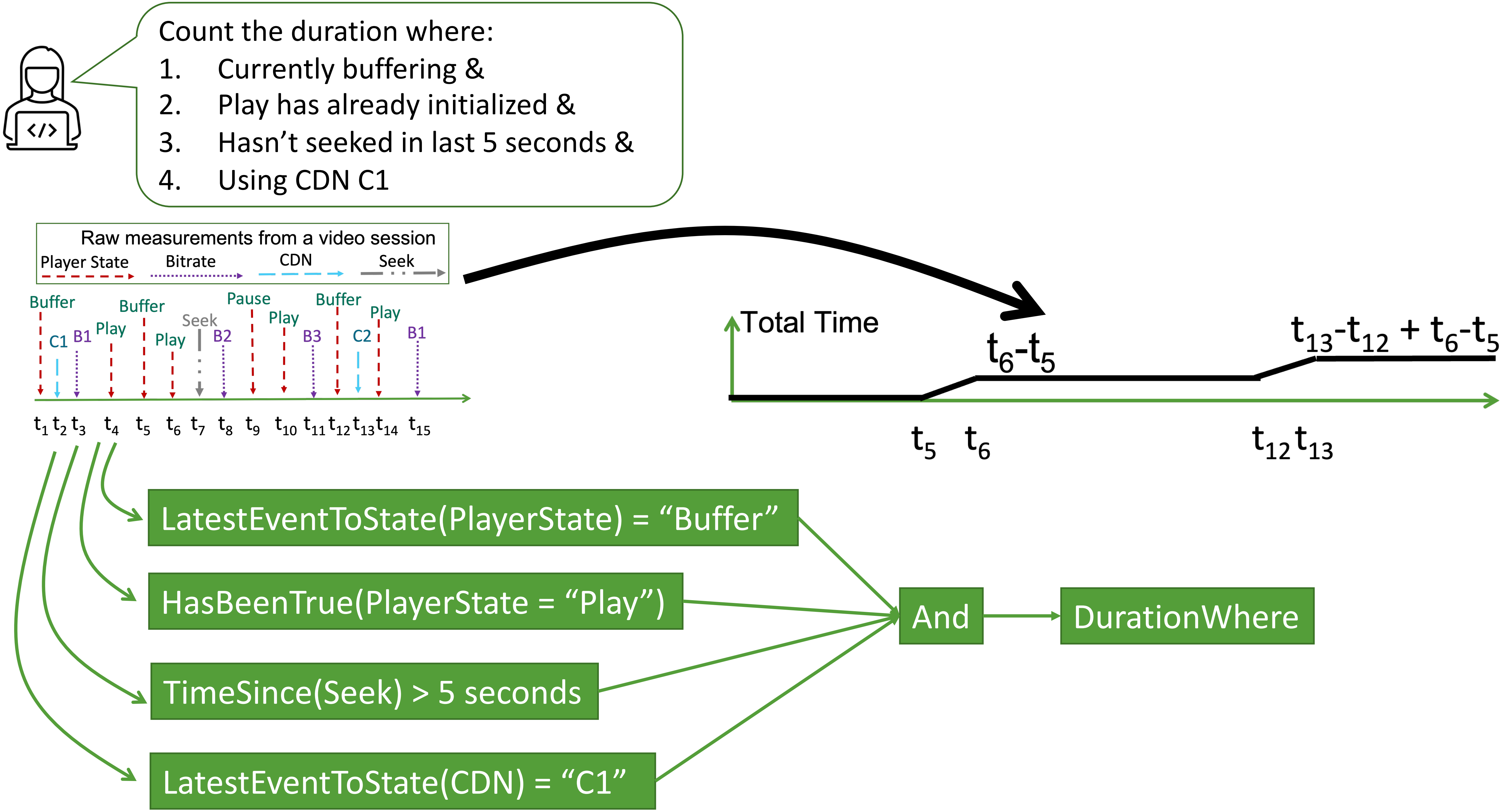
Operator: TimeSince(Seek) > 5 seconds

Whiteboarding: Connection-Induced Rebuffering w/ C1

Count the duration where:

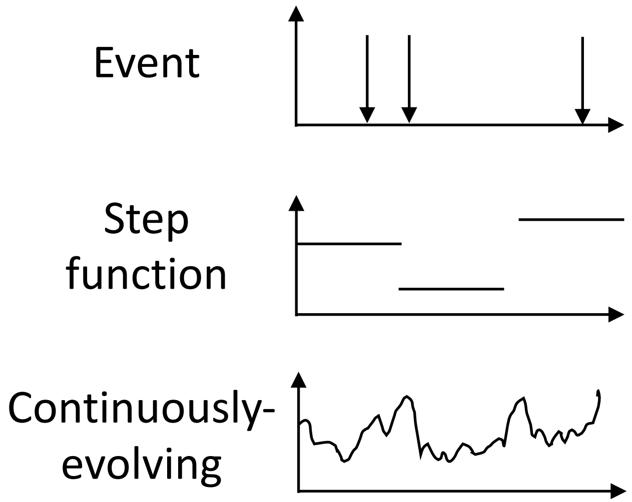
1. Currently buffering &
2. Play has already initialized &
3. Hasn't seeked in last 5 seconds **&**
4. Using CDN C1



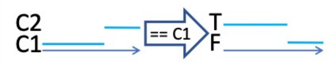
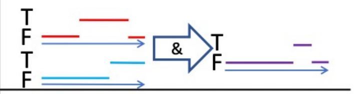
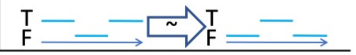
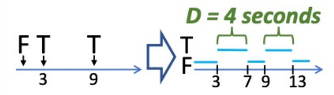
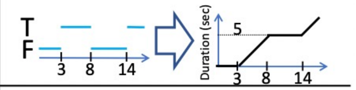
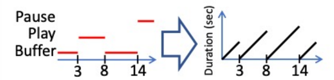

Count the duration where:
1. Currently buffering &
2. Play has already initialized &
3. Hasn't seeked in last 5 seconds &
4. Using CDN C1
Raw measurements from a video session
Player State
Bitrate
CDN
Seek
Buffer
C1
B1
Play
Buffer
Play
Seek
B2
Pause
Play
B3
Buffer
C2
Play
B1
t1 t2 t3 t4 t5 t6 t7 t8 t9 t10 t11 t12 t13 t14 t15
Total Time
t6-t5
t13-t12 + t6-t5
t5 t6
t12 t13
LatestEventToState(PlayerState) = "Buffer"
HasBeenTrue(PlayerState = "Play")
TimeSince(Seek) > 5 seconds
LatestEventToState(CDN) = "C1"
And
DurationWhere

Timeline, in a nutshell

Data abstraction with 3 types of timeline dynamics

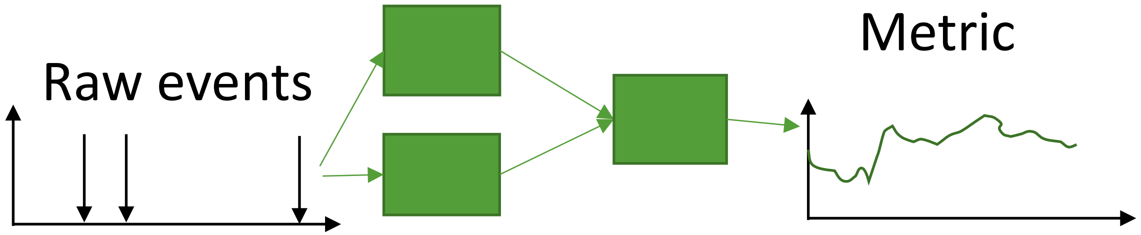


Library of operators

Timeline generalizations of classical Operators		
==, <, > [constant]	Compare each update or state with a fixed value, producing True or False	
&, | [timeline]	Combine 2 timelines by applying a logical operation at each point in time	
~	Logically invert each update or state	
Timeline-specific Operators		
TL_HasExisted	A StateDynamics timeline of the *cumulative OR*	
TL_HasExistedWithin	As TL_HasExisted, but resets to False after a specified duration *D* without True values	 D = 4 seconds
TL_LatestEventToState	A StateDynamics Timeline of the latest update	
TL_DurationWhere	A Numerical Timeline of the cumulative duration where the state was True	
TL_DurationInCurState	A Numerical Timeline of the duration since the last state change	

...

Compositional language for defining DAG of operators



Connectors with external data sources/sinks

Outline for talk

- What is Time-State Analytics
- Time-State Analytics not well supported by status quo
- Introducing the Timeline abstraction
- **Early Promise + Next Steps**

Timeline Reduced Dev Effort at Conviva

```
WITH SeekAsPlayerState(T, P) as (
  SELECT T, P FROM heartbeats WHERE P IS NOT NULL
  UNION SELECT T, "Seek_st" FROM heartbeats WHERE A IS NOT NULL
  UNION SELECT T + 5, "Seek_ed" FROM heartbeats WHERE A IS NOT NULL ),
IgnoreBufBeforePlay(T, P) as (
  SELECT T, P FROM (
    SELECT T, P, Max(If(P == 'play', 1, 0)) OVER (PARTITION BY 1 ORDER BY T)
      ↪ as H
    FROM SeekAsPlayerState) WHERE H == True ),
DuringBufferTable(T, P, DB) as (
  SELECT T, P, LAST(tmp1) IGNORE NULLS OVER (PARTITION BY 1 ORDER BY T)
  FROM (
    SELECT T, P,
          CASE P WHEN 'buffer' THEN True WHEN 'Seek_st' THEN NULL WHEN 'Seek_ed'
            ↪ THEN NULL ELSE FALSE END as tmp1
    From IgnoreBufBeforePlay ) ),
DuringSeekTable(T, P, DB, DS) as (
  SELECT T, P, DB,
          (T - Max(If(P == 'Seek_st', T, 0)) OVER (PARTITION BY 1 ORDER BY T)
            ↪ ) < 5 as tmp2
  FROM DuringBufferTable ),
IgnoreBufInSeek(T, P) as (
  SELECT T, P FROM (
    SELECT T, DS, IF(P == 'Seek_ed' and DB, 'buffer', P) as P
   FROM DuringSeekTable ) WHERE NOT (P == 'buffer' AND DS) ),
WithCDNAndQuery(T, P, C) as (
 SELECT T, P, NULL FROM IgnoreBufInSeek
 UNION SELECT T, NULL, C FROM heartbeats where C IS NOT NULL
 UNION SELECT 2022-07-21 10:05, NULL, NULL s),
Intervals(Ed, St, State, CDN) as (
  SELECT T, LEAD(T, 1) OVER (PARTITION BY 1 ORDER BY T), P, C
  FROM (
    SELECT T,
          LAST(P) IGNORE NULLS OVER (PARTITION BY 1 ORDER BY T) as P,
          LAST(C) IGNORE NULLS OVER (PARTITION BY 1 ORDER BY T) as C
    FROM WithCDNAndQuery ) )
SELECT SUM(St - Ed) as result FROM Intervals
WHERE Ed < 2022-07-21 10:05 AND State == 'buffer' AND CDN == 'CDN1'
```

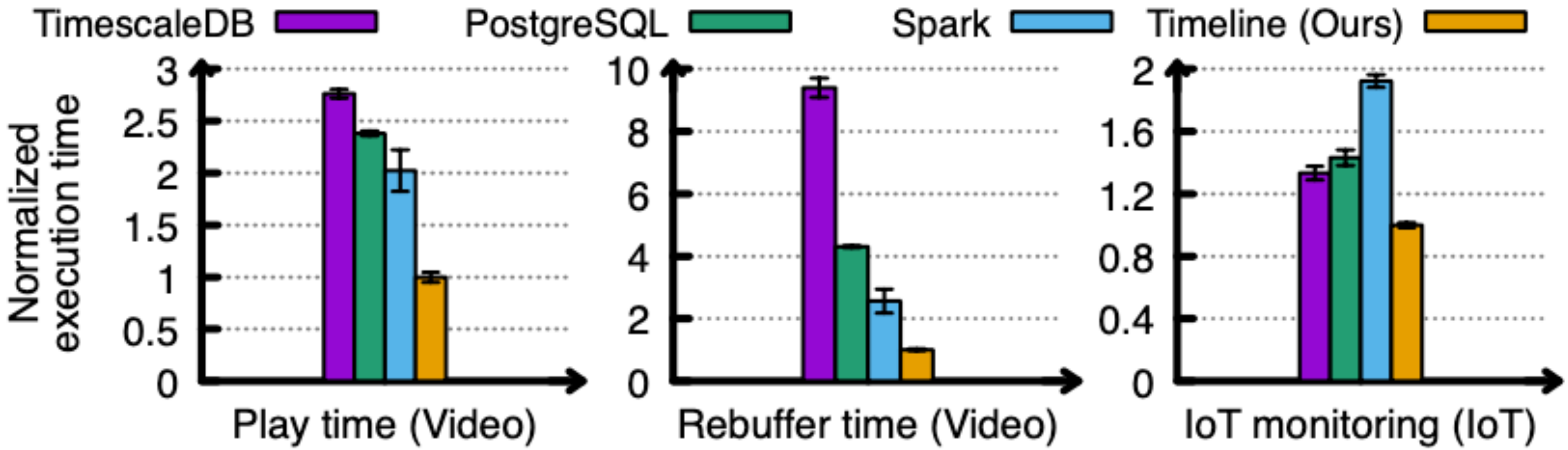
Onboarding: Weeks → Days

Semantic Bugs: Dropped by 80%

```
SELECT TL_DurationWhere(
  TL_LatestEventToState(playerStateChange) = 'buffer' AND
  TL_HasExisted(playerStateChange = 'play') AND
  NOT TL_HasExistedWithin(userAction = 'seek', 5s) AND
  TL_LatestEventToState(cdnChange) = 'CDN1'
) AS result
FROM heartbeats
TIMELINE WITH EVENT TIME t
EVALUATE AT EVENT TIME 2022-07-21 10:05:00
```

Prototype query language

Timeline Offers Reduced Cost



2-10X Faster Execution Time!

Future Outlook

- Applications to many domains
  - Cybersecurity, IoT, logistics, manufacturing, …
- Visual interfaces to democratize TSA
- Even better performance
- Streaming implementation

Takeaways

- Growing need for *Time-State Analytics* across different domains
- Fundamentally hard problem:
  Stateful, Context-Sensitive, Continuous
- State-of-art systems (e.g., streaming systems, data warehouses, RDBMS) ill suited
  - Why: Classical tabular model for data processing is ineffective for Time-State Analytics
    - Great for simple stateless filter/aggregation but not Time-State Analytics
    - High cost, low performance + High dev effort, many bugs
- Our work: Timeline → A *geometric abstraction* for Time-State Analytics
  - Early promise: Up to 10X better cost/performance AND 10X reduced effort
  - New opportunities: Generality, No-code Intents