DEBS Grand Challenge: RDF Stream Processing with CQELS Framework for Real-time Analysis

Danh Le-Phuoc Minh Dao-Tran Anh Le Tuan
Manh Nguyen Duc Manfred Hauswirth

The 9th ACM International Conference on Distributed Event-Based Systems

Oslo, June 30, 2015

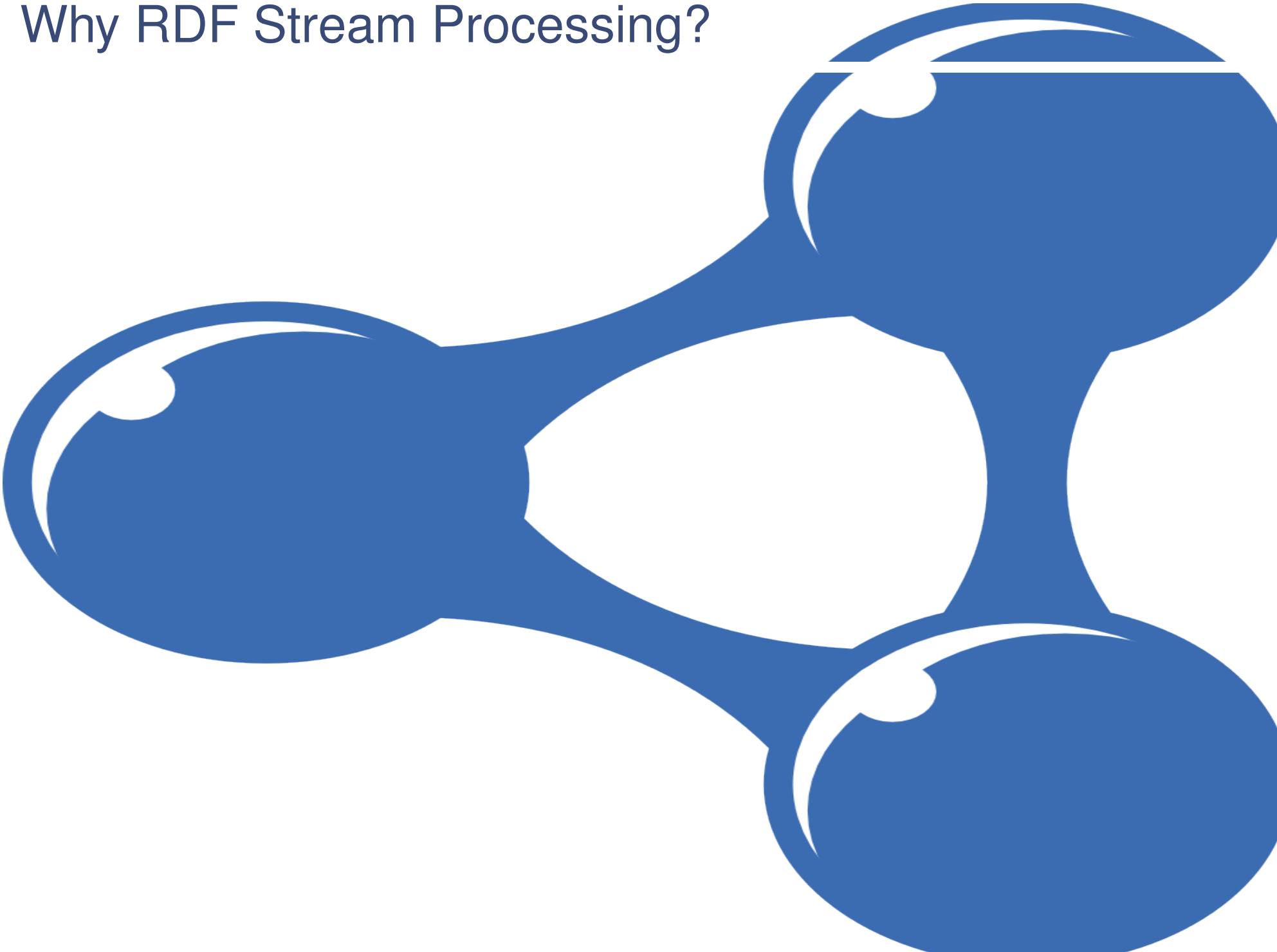
NUI Galway
OÉ Gaillimh

Insight

FAKULTÄT
FÜR !NFORMATIK
Faculty of Informatics

Technische Universität Berlin

Why RDF Stream Processing?



Interoperability

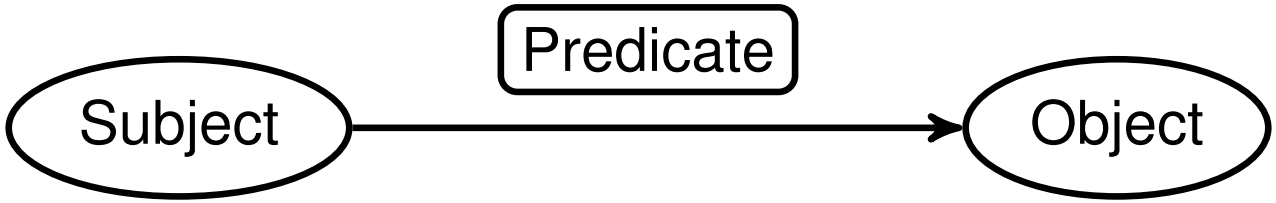
Interoperability

Interoperability

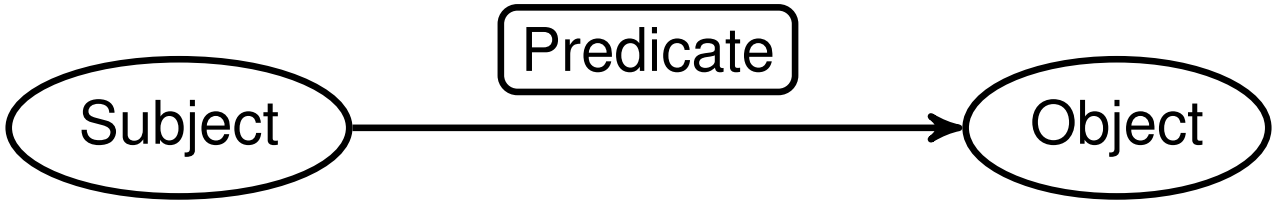
Interoperability

RDF

RDF

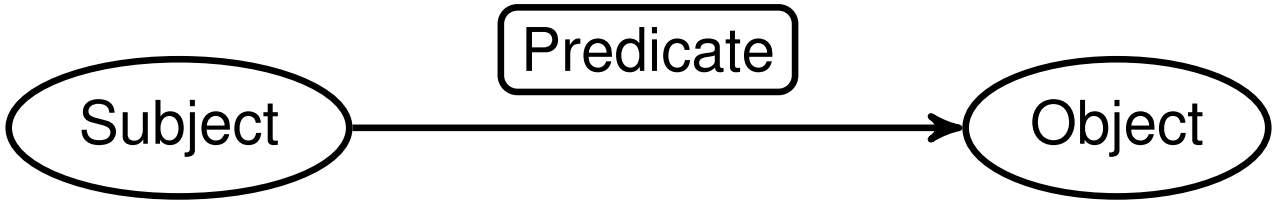


RDF




```
07290D3599E7A0D62097A346EFCC1FB5,
2013-01-01 00:00:00,2013-01-01 00:02:00,
-73.956528,40.716976,-73.962440,40.715008, 3.50,0.00
```

RDF




```
07290D3599E7A0D62097A346EFCC1FB5,
2013-01-01 00:00:00,2013-01-01 00:02:00,
-73.956528,40.716976,-73.962440,40.715008, 3.50,0.00
```

$$
\left\{
\begin{array}{l}
:trip_1\ :taxi\ \texttt{"07290D3599E7A0D62097A346EFCC1FB5"}. \\
:trip_1\ :pickup\_datetime\ \texttt{"2013-01-01 00:00:00"}. \\
:trip_1\ :dropoff\_datetime\ \texttt{"2013-01-01 00:02:00"}. \\
:trip_1\ :pickLon\ -73.956528. \\
:trip_1\ :pickLat\ 40.716976. \\
:trip_1\ :dropLon\ -73.962440. \\
:trip_1\ :dropLat\ 40.715008. \\
:trip_1\ :fare\ 3.5. \\
:trip_1\ :tip\ 0.0.
\end{array}
\right\}
$$

SPARQL

```
SELECT


FROM
WHERE
```

SPARQL

```
SELECT


FROM <http://example/taxi.rdf>
WHERE
```

SPARQL

```
SELECT


FROM <http://example/taxi.rdf>
WHERE {
    ?trip :pickLon ?pLon.  ?trip :pickLat ?pLat.
    ?trip :dropLon ?dLon.  ?trip :dropLat ?dLat.
}
```

SPARQL

```
SELECT (ROUND((41.474937-?pLat)/0.005986)     AS ?pE)
       (ROUND((74.913585+?pLon)/0.004491556) AS ?pS)
       (ROUND((41.474937-?dLat)/0.005986)     AS ?dE)
       (ROUND((74.913585+?dLon)/0.004491556) AS ?dS)

FROM <http://example/taxi.rdf>
WHERE {
    ?trip :pickLon ?pLon.  ?trip :pickLat ?pLat.
    ?trip :dropLon ?dLon.  ?trip :dropLat ?dLat.
}
```

SPARQL

```
SELECT (ROUND((41.474937-?pLat)/0.005986)     AS ?pE)
       (ROUND((74.913585+?pLon)/0.004491556) AS ?pS)
       (ROUND((41.474937-?dLat)/0.005986)     AS ?dE)
       (ROUND((74.913585+?dLon)/0.004491556) AS ?dS)

FROM <http://example/taxi.rdf>
WHERE {
    ?trip :pickLon ?pLon.  ?trip :pickLat ?pLat.
    ?trip :dropLon ?dLon.  ?trip :dropLat ?dLat.
}

HAVING  (?pE>0 && ?pE<301 && ?pS>0 && ?pS<301 &&
         ?dE>0 && ?dE<301 && ?dS>0 && ?dS<301)
```

SPARQL

```
SELECT (ROUND((41.474937-?pLat)/0.005986)     AS ?pE)
       (ROUND((74.913585+?pLon)/0.004491556) AS ?pS)
       (ROUND((41.474937-?dLat)/0.005986)     AS ?dE)
       (ROUND((74.913585+?dLon)/0.004491556) AS ?dS)
       (COUNT(?trip) AS ?freq)
FROM <http://example/taxi.rdf>
WHERE {
    ?trip :pickLon ?pLon.  ?trip :pickLat ?pLat.
    ?trip :dropLon ?dLon.  ?trip :dropLat ?dLat.
}
GROUP BY ?pE ?pS ?dE ?dS
HAVING  (?pE>0 && ?pE<301 && ?pS>0 && ?pS<301 &&
         ?dE>0 && ?dE<301 && ?dS>0 && ?dS<301)
```

SPARQL

```
SELECT (ROUND((41.474937-?pLat)/0.005986)     AS ?pE)
       (ROUND((74.913585+?pLon)/0.004491556) AS ?pS)
       (ROUND((41.474937-?dLat)/0.005986)     AS ?dE)
       (ROUND((74.913585+?dLon)/0.004491556) AS ?dS)
       (COUNT(?trip) AS ?freq)
FROM <http://example/taxi.rdf>
WHERE {
    ?trip :pickLon ?pLon.  ?trip :pickLat ?pLat.
    ?trip :dropLon ?dLon.  ?trip :dropLat ?dLat.
}
GROUP BY ?pE ?pS ?dE ?dS
HAVING  (?pE>0 && ?pE<301 && ?pS>0 && ?pS<301 &&
         ?dE>0 && ?dE<301 && ?dS>0 && ?dS<301)
ORDER BY ?freq
LIMIT 10
```

SPARQL

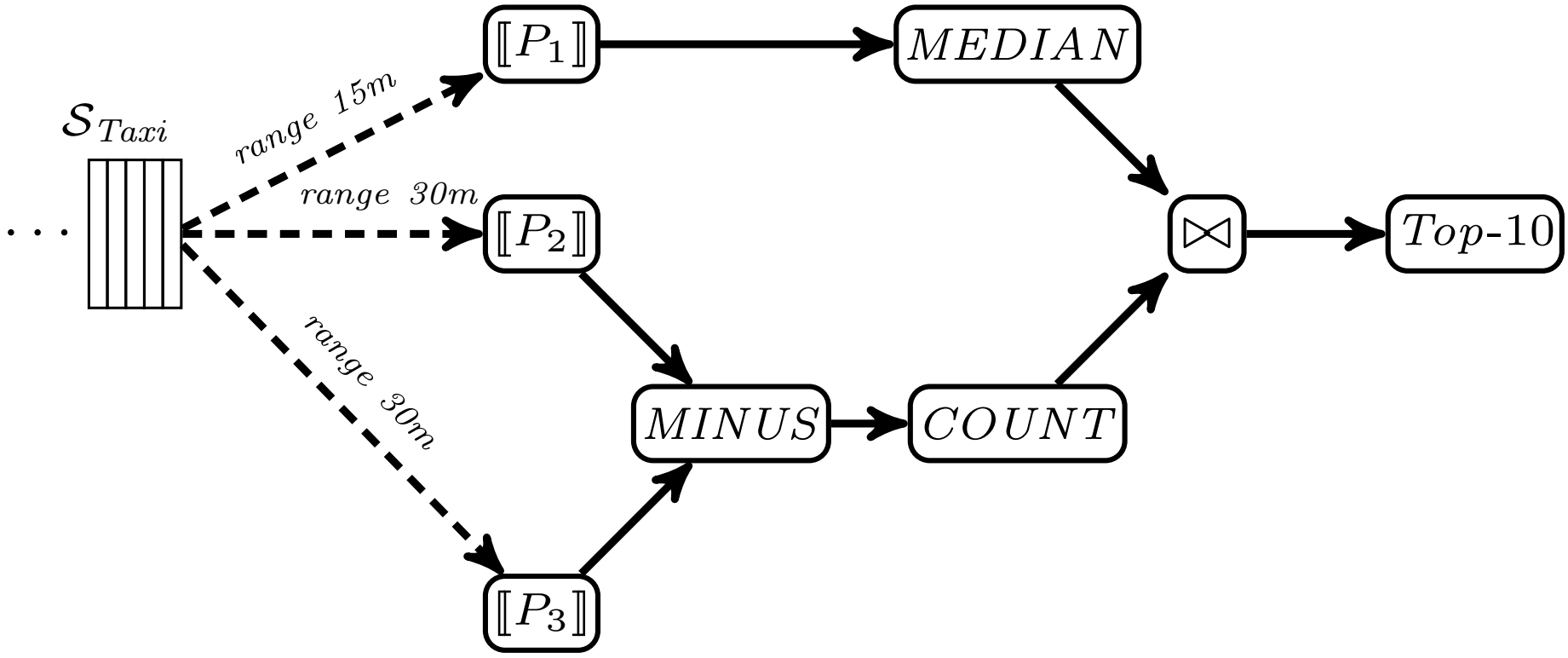
```
SELECT (ROUND((41.474937-?pLat)/0.005986)     AS ?pE)
       (ROUND((74.913585+?pLon)/0.004491556) AS ?pS)
       (ROUND((41.474937-?dLat)/0.005986)     AS ?dE)
       (ROUND((74.913585+?dLon)/0.004491556) AS ?dS)
       (COUNT(?trip) AS ?freq)
FROM <http://example/taxi.rdf>
WHERE {
    ?trip :pickLon ?pLon.  ?trip :pickLat ?pLat.
    ?trip :dropLon ?dLon.  ?trip :dropLat ?dLat.
}
GROUP BY ?pE ?pS ?dE ?dS
HAVING  (?pE>0 && ?pE<301 && ?pS>0 && ?pS<301 &&
         ?dE>0 && ?dE<301 && ?dS>0 && ?dS<301)
ORDER BY ?freq
LIMIT 10
```

$\{\{\texttt{?pE} \mapsto 127, \texttt{?pS} \mapsto 213, \texttt{?dE} \mapsto 127, \texttt{?dS} \mapsto 212, \texttt{?freq} \mapsto 1\}\}$.

RSP Queries in CQELS

```
SELECT (ROUND((41.474937-?pLat)/0.005986)     AS ?pE)
       (ROUND((74.913585+?pLon)/0.004491556) AS ?pS)
       (ROUND((41.474937-?dLat)/0.005986)     AS ?dE)
       (ROUND((74.913585+?dLon)/0.004491556) AS ?dS)
       (COUNT(?trip) AS ?freq)
WHERE {
  STREAM <Taxi> [RANGE 30 minutes] {
    ?trip :pickLon ?pLon.  ?trip :pickLat ?pLat.
    ?trip :dropLon ?dLon.  ?trip :dropLat ?dLat.
  }
}
GROUP BY ?pE ?pS ?dE ?dS
HAVING  (?pE>0 && ?pE<301 && ?pS>0 && ?pS<301 &&
         ?dE>0 && ?dE<301 && ?dS>0 && ?dS<301)
ORDER BY ?freq
LIMIT 10
```

Query Plan of Q_2



CQELS Solutions for the Grand Challenge

CQELS Solutions for the Grand Challenge

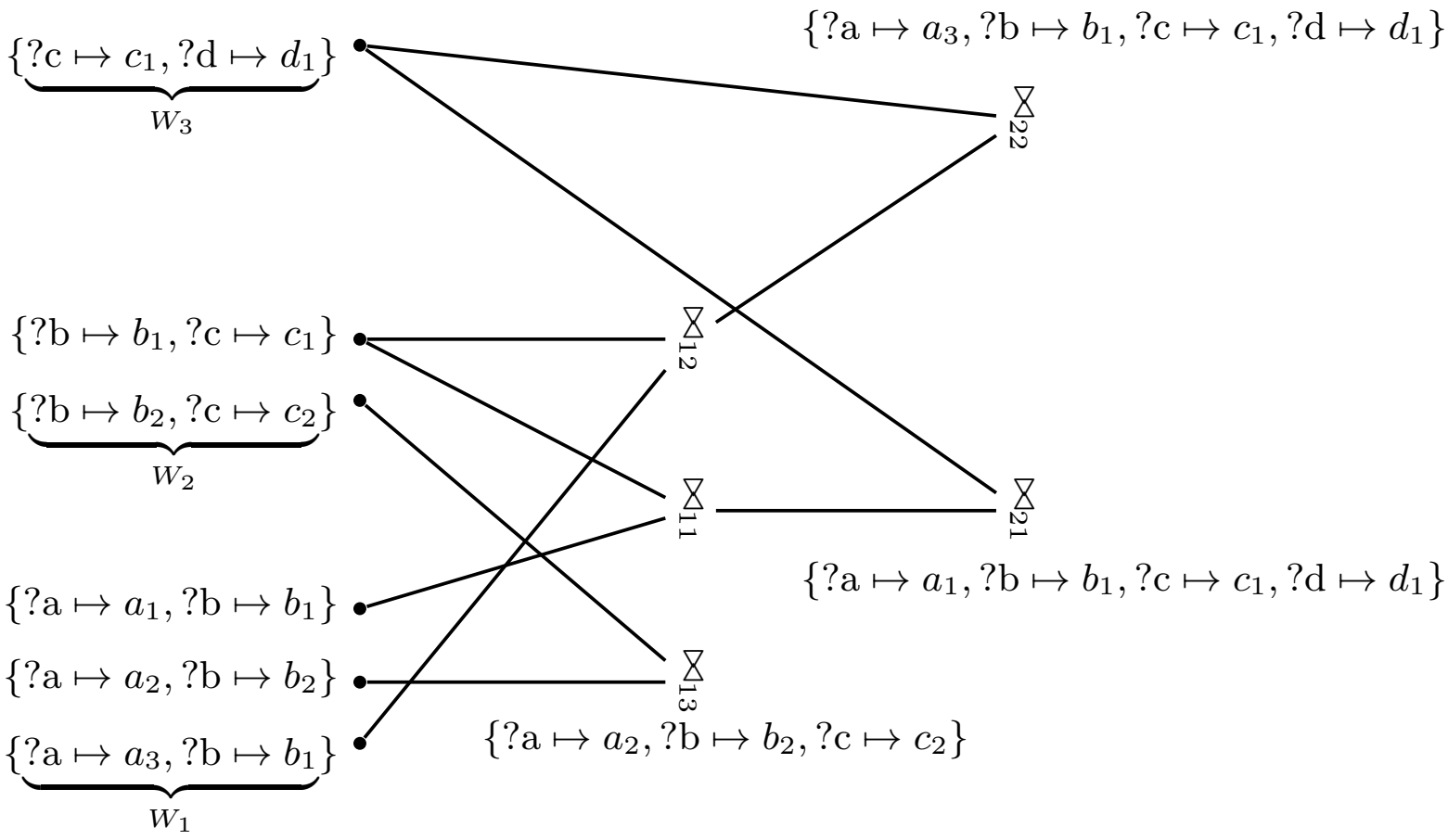
High Performance Data Structures

CQELS Solutions for the Grand Challenge

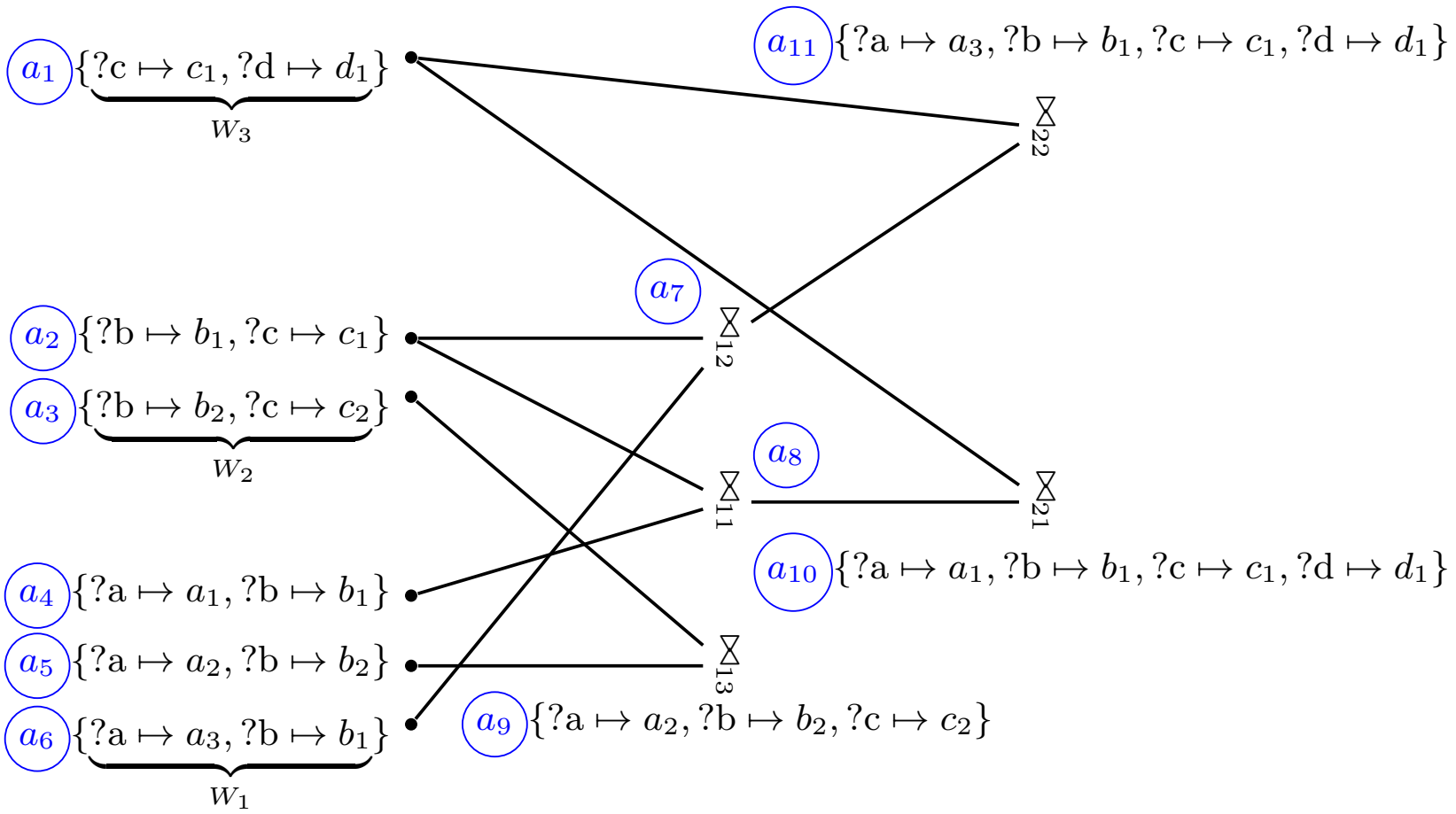
High Performance Data Structures

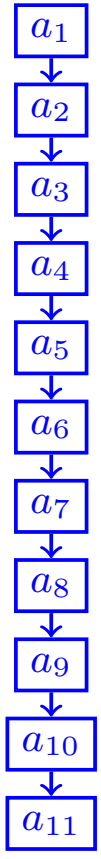
Incremental Algorithms

Tree-based Data Structures

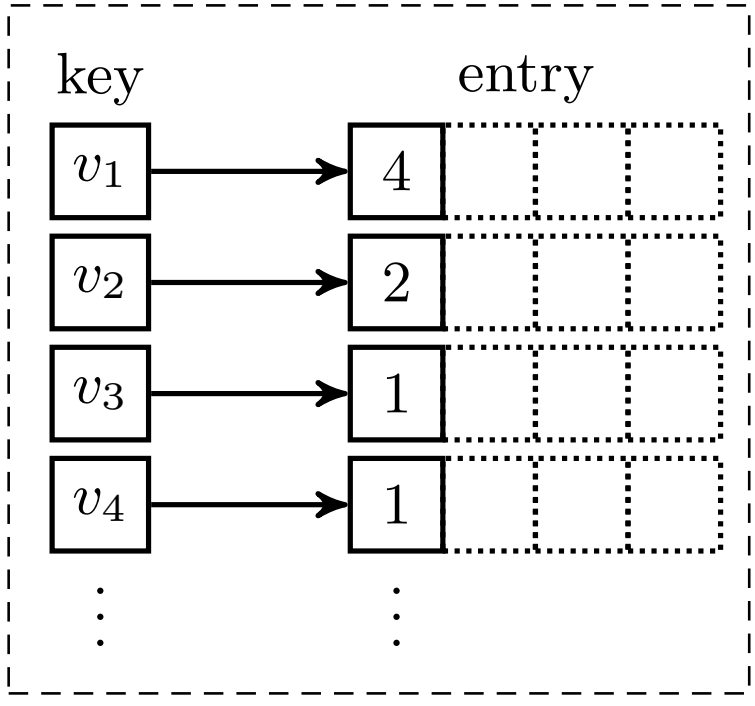


Input Buffers

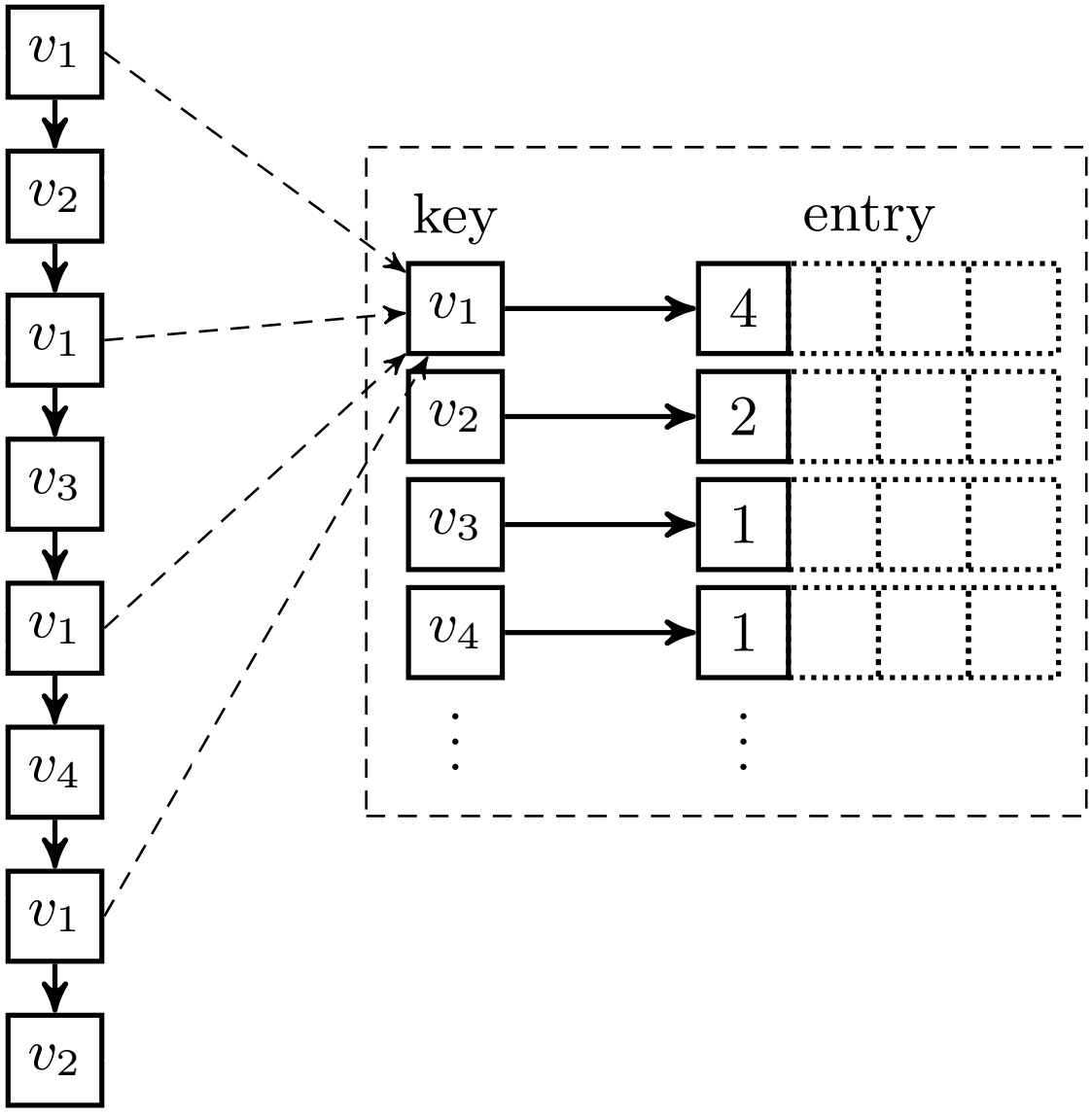




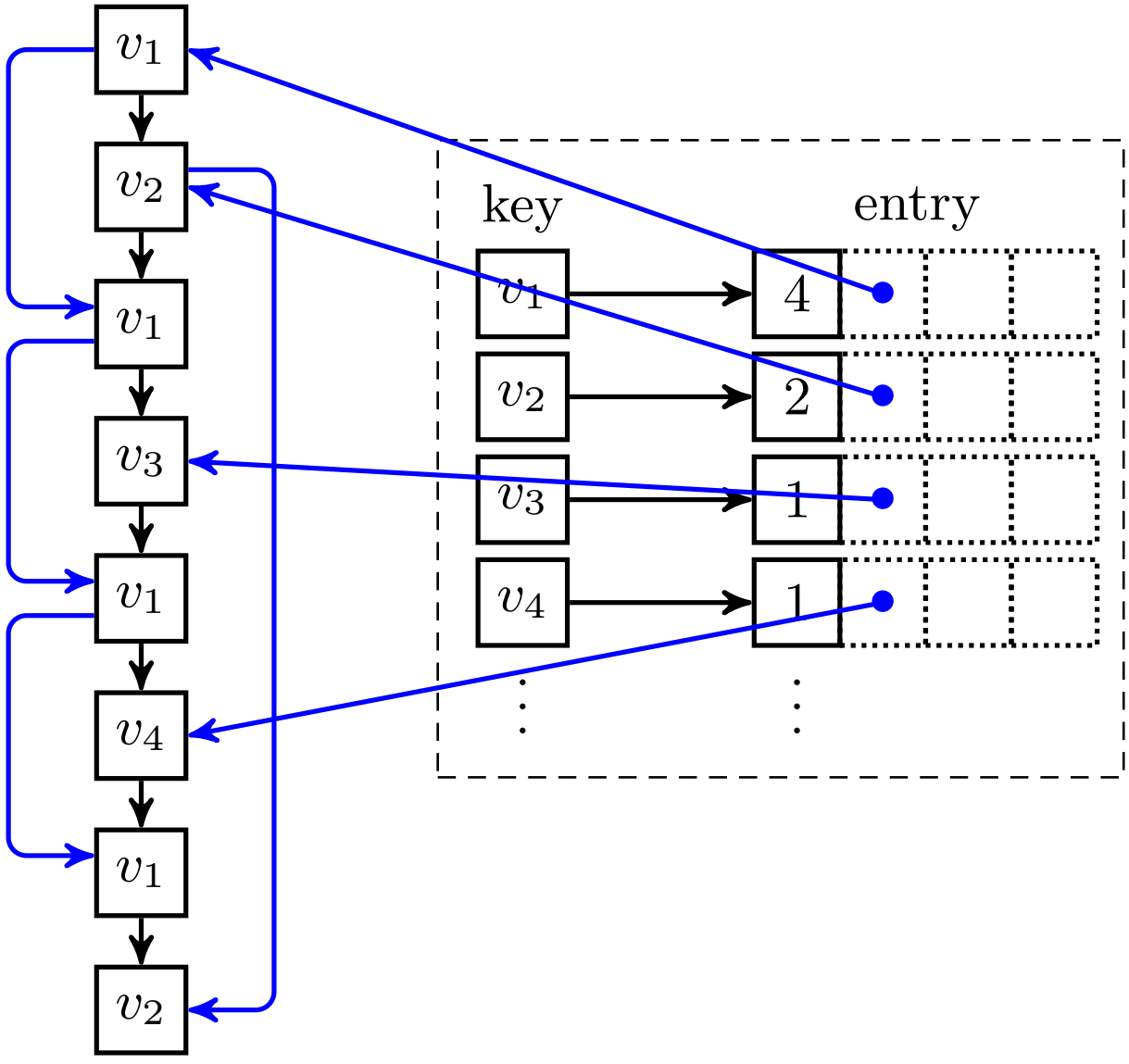
Indexed Buffers



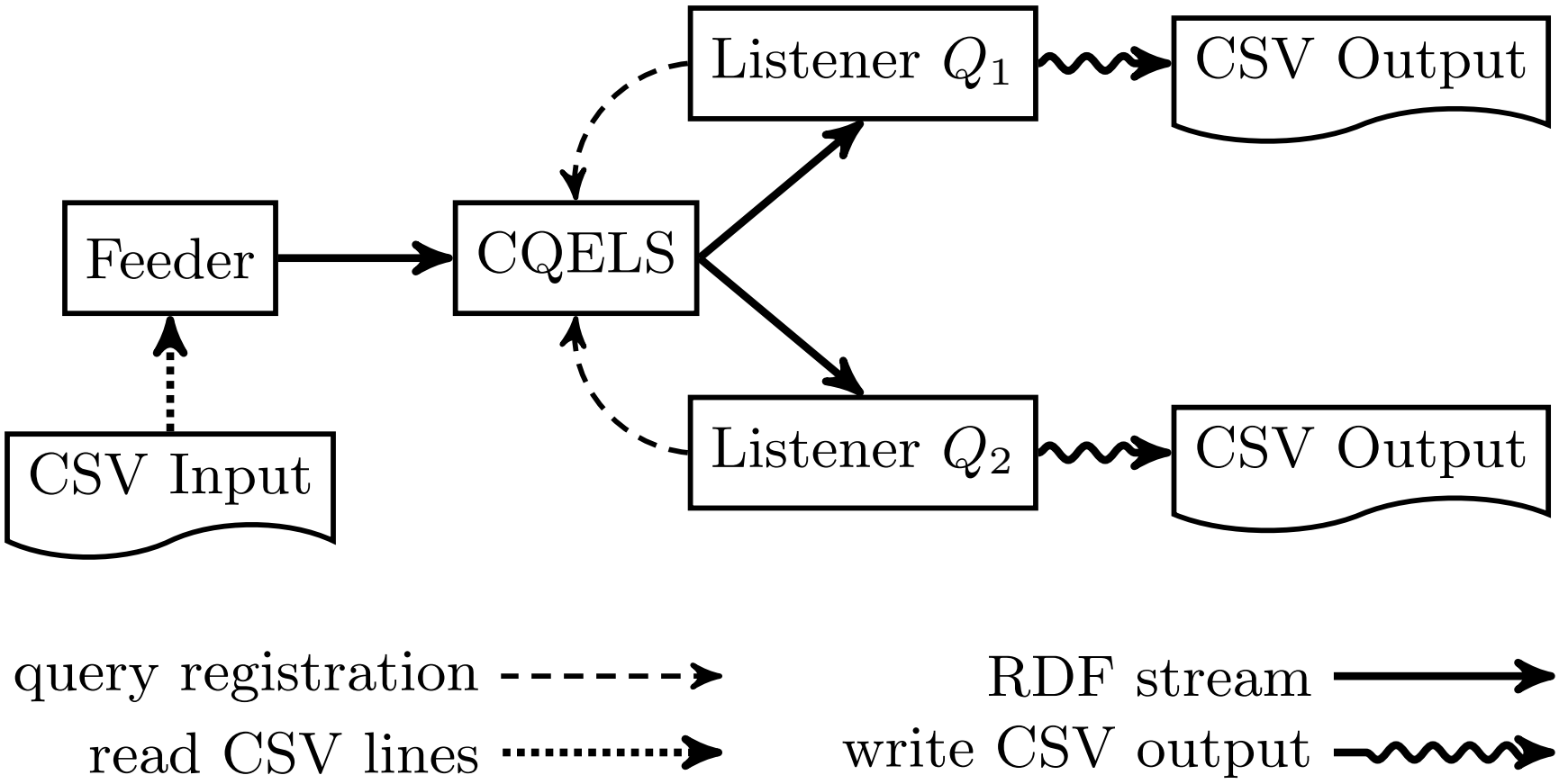
One-way Index



Ring Index



Deployment and Tests



Compare CQELS to a Base-line System

	Avg. delay time (ms)				Exe. time (s)		
	Indv.		Mixed		Indv.		Mixed
	Q_1	Q_2	Q_1	Q_2	Q_1	Q_2	Q_1 & Q_2
C	0.022	0.068	0.023	0.075	65.77	69.02	76.64
B	5	194	174	220	5000	22000	28000

Delay time (per output):
right after reading the input $\rightarrow$ right before writing the output

Execution time:
first input line read $\rightarrow$ final output streamed out

Compare CQELS to a Base-line System

	Avg. delay time (ms)				Exe. time (s)		
	Indv.		Mixed		Indv.		Mixed
	Q_1	Q_2	Q_1	Q_2	Q_1	Q_2	Q_1 & Q_2
C	0.022	0.068	0.023	0.075	65.77	69.02	76.64
B	5	194	174	220	5000	22000	28000

Delay time (per output):
right after reading the input $\rightarrow$ right before writing the output

Execution time:
first input line read $\rightarrow$ final output streamed out

Exp: vary window sizes

Compare CQELS to a Base-line System

	Avg. delay time (ms)				Exe. time (s)		
	Indv.		Mixed		Indv.		Mixed
	Q_1	Q_2	Q_1	Q_2	Q_1	Q_2	Q_1 & Q_2
C	0.022	0.068	0.023	0.075	65.77	69.02	76.64
B	5	194	174	220	5000	22000	28000

Delay time (per output):
right after reading the input $\rightarrow$ right before writing the output

Execution time:
first input line read $\rightarrow$ final output streamed out

Exp: vary window sizes $\rightarrow$ no change in average delay time!

Conclusions

RDF Stream Processing for the DEBS 2015 Grand Challenge

Techniques implemented in CQELS

- High performance data structures
- Incremental algorithms (details in paper)

Experimental results

- Comparison to a base-line system
- Experiment on varying window sizes