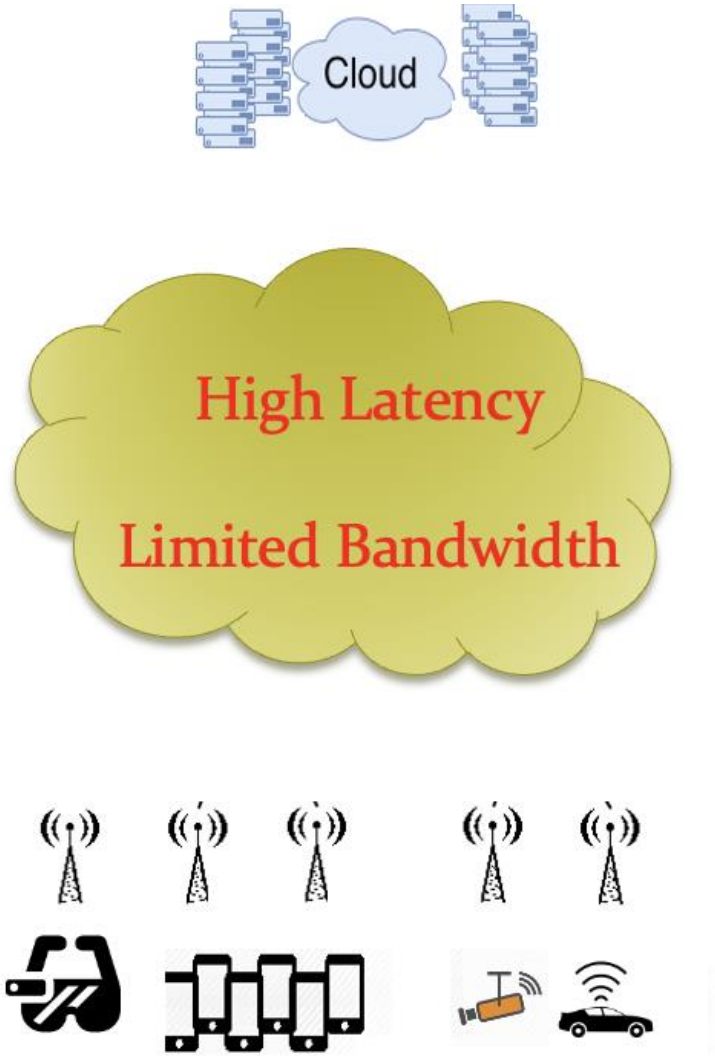
# Efficient Data Processing for Edge Computing

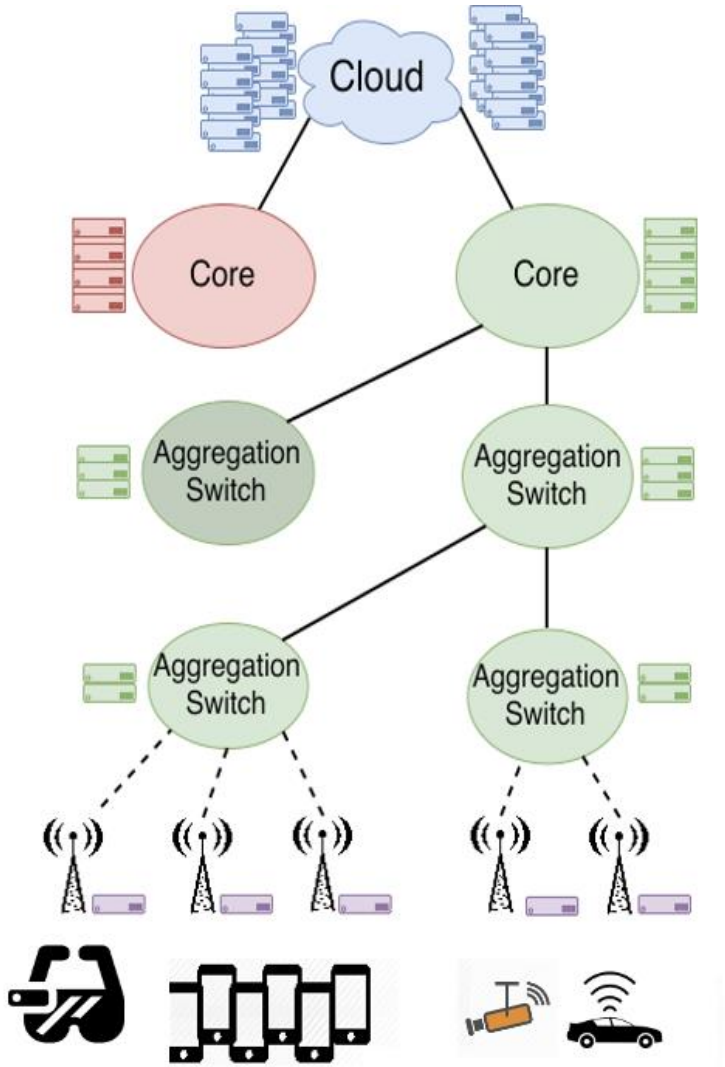
*Oana Balmau, McGill University*

*Dec 17, 2024, Women in CS Workshop Cameroon*

# Cloud Applications Today



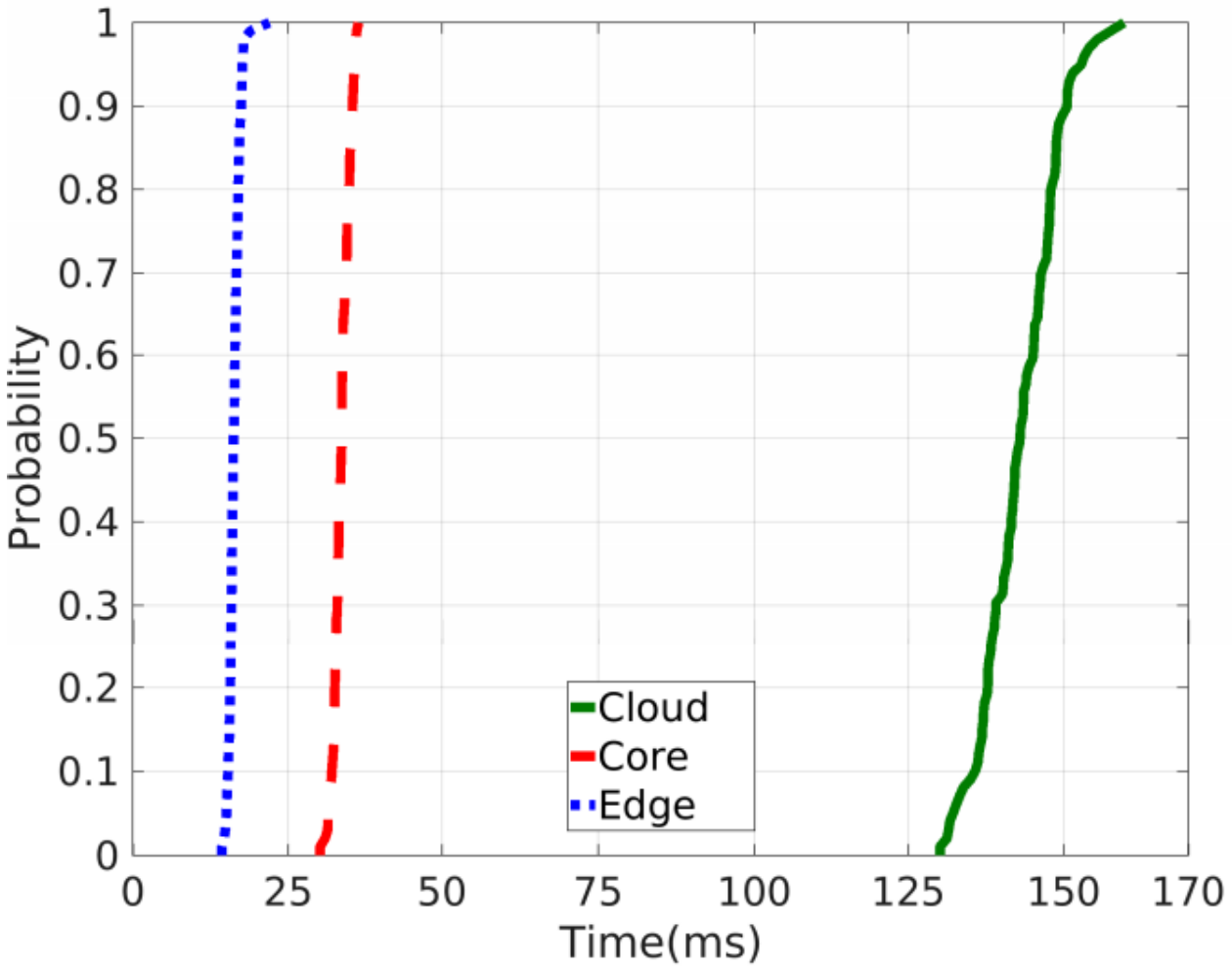
# Hierarchical Cloud




Example:

- AWS Region
- AWS Local
- AWS Wavelength

# License Plate Recognition

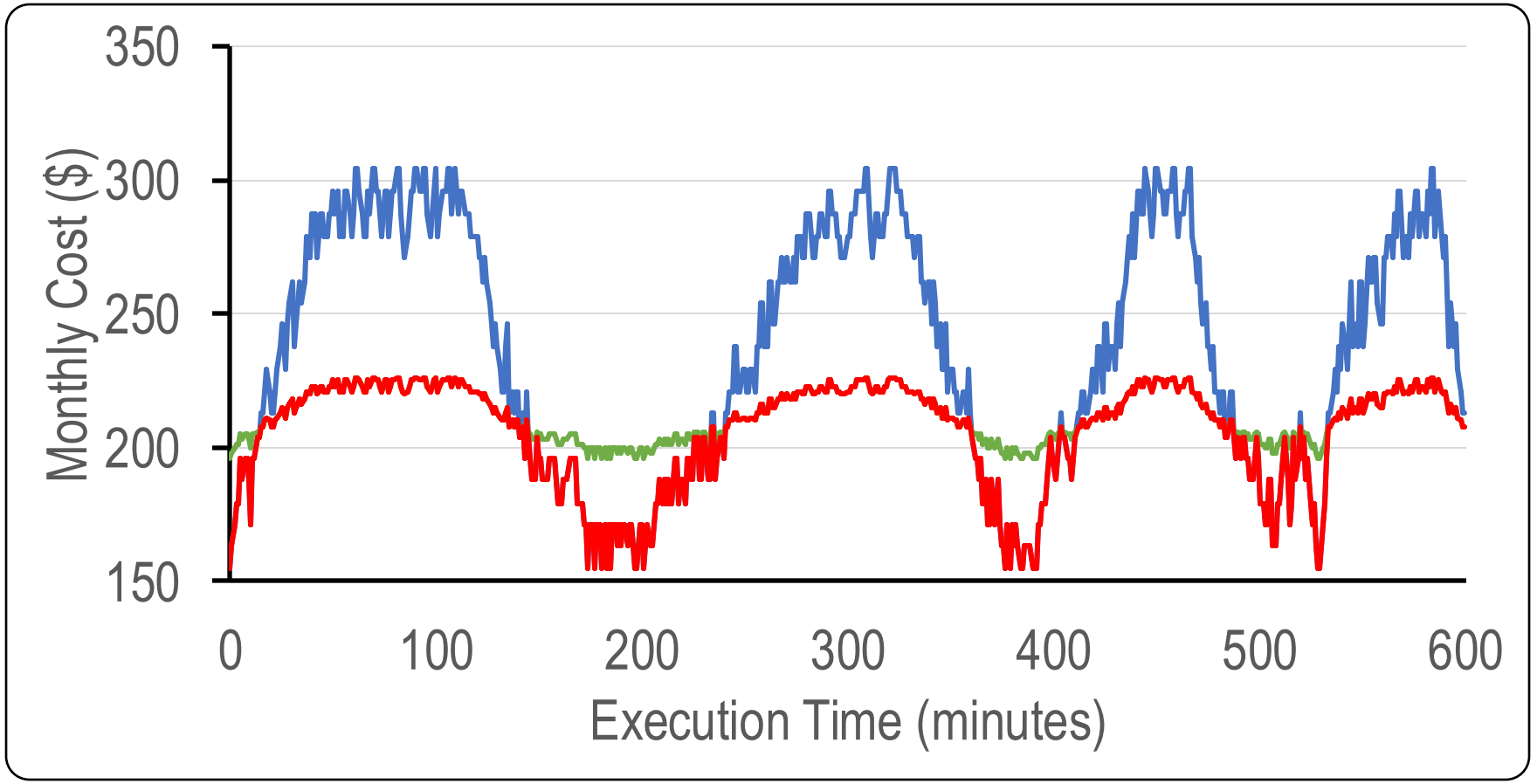


# Need for Adaptation – Variable Workloads

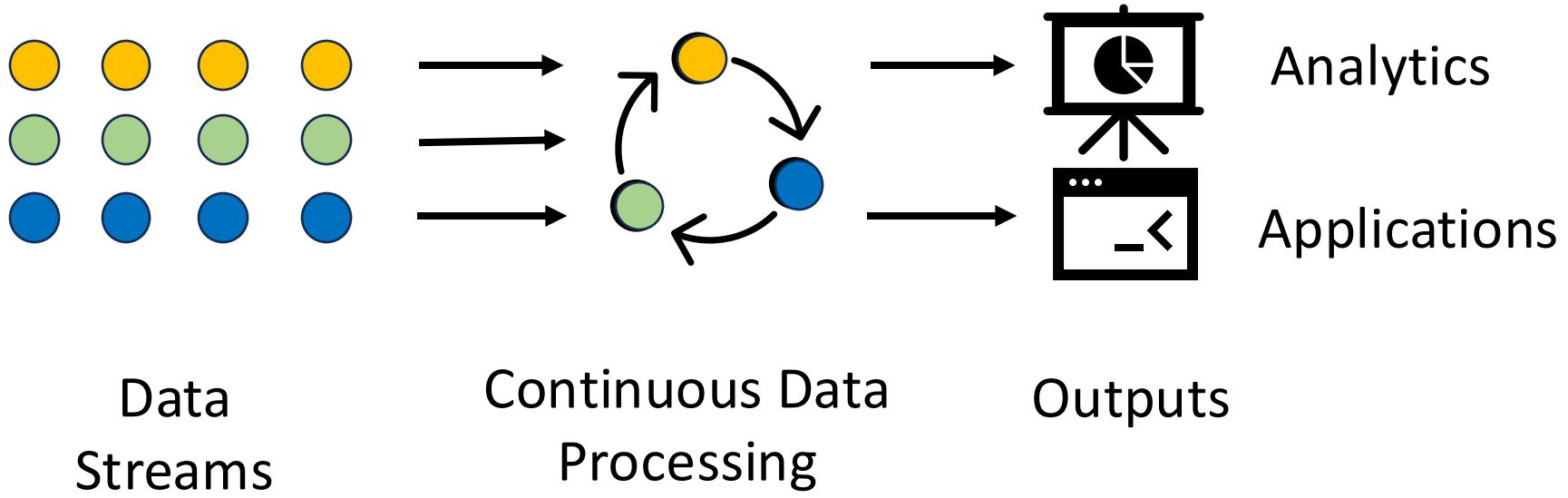
Cloud Only
- Compute cheaper, bandwidth expensive

Edge Only
- Compute expensive, bandwidth cheaper

Adaptive
- Balance cost between compute and bandwidth



# What Is Stream Processing?



# Stream Processing Is Popular





Popular Stream Processing Frameworks


Real Time Analysis


Optimize search rankings in real time

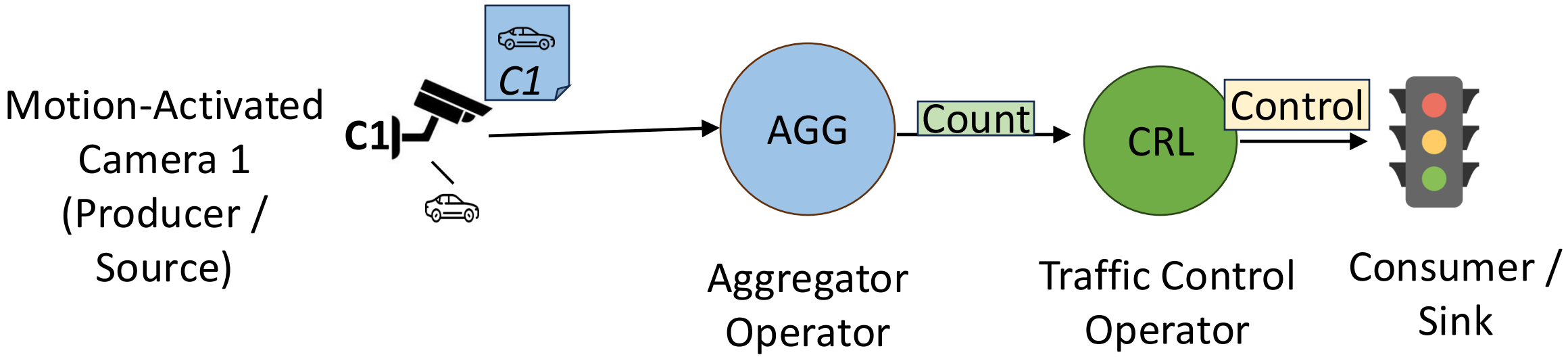

Process and analyze streaming data


Real-time monitoring

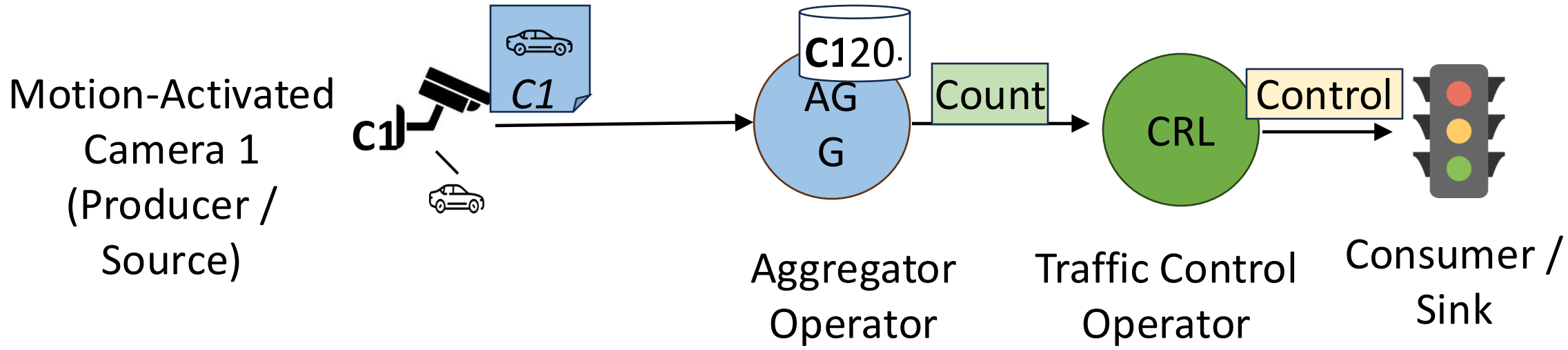
Organizations using these frameworks[1]

1. https://flink.apache.org/what-is-flink/powered-by/

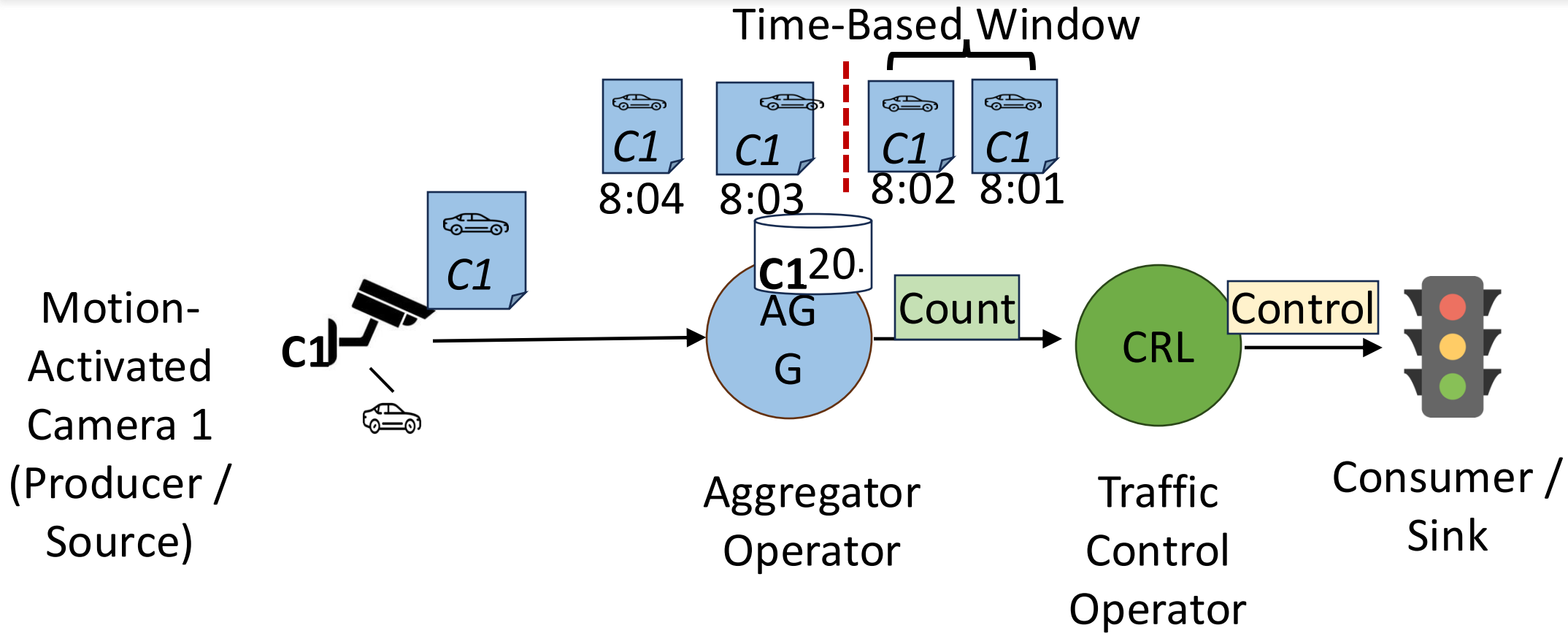
# Stateful Stream Processing - Basics

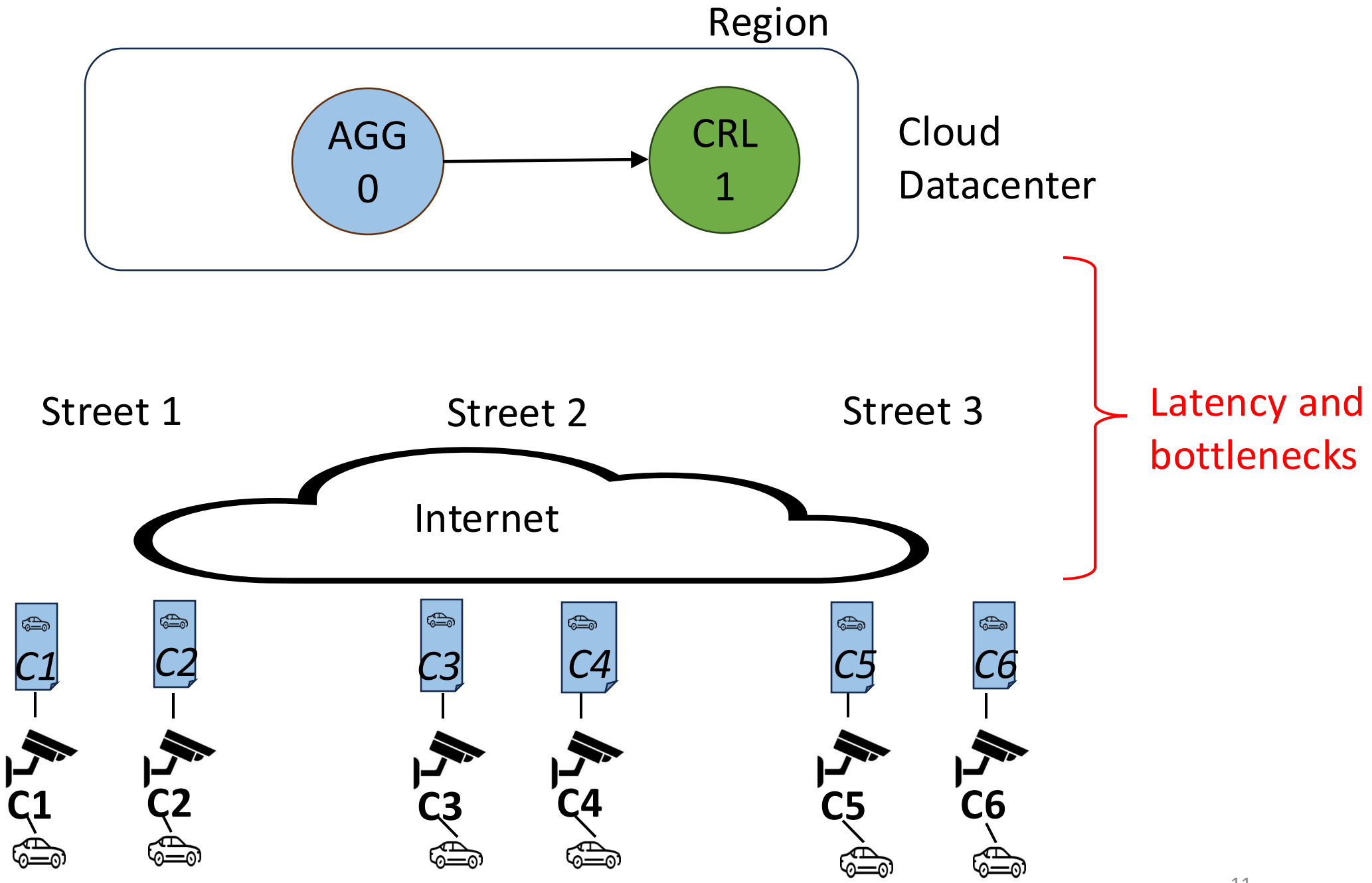


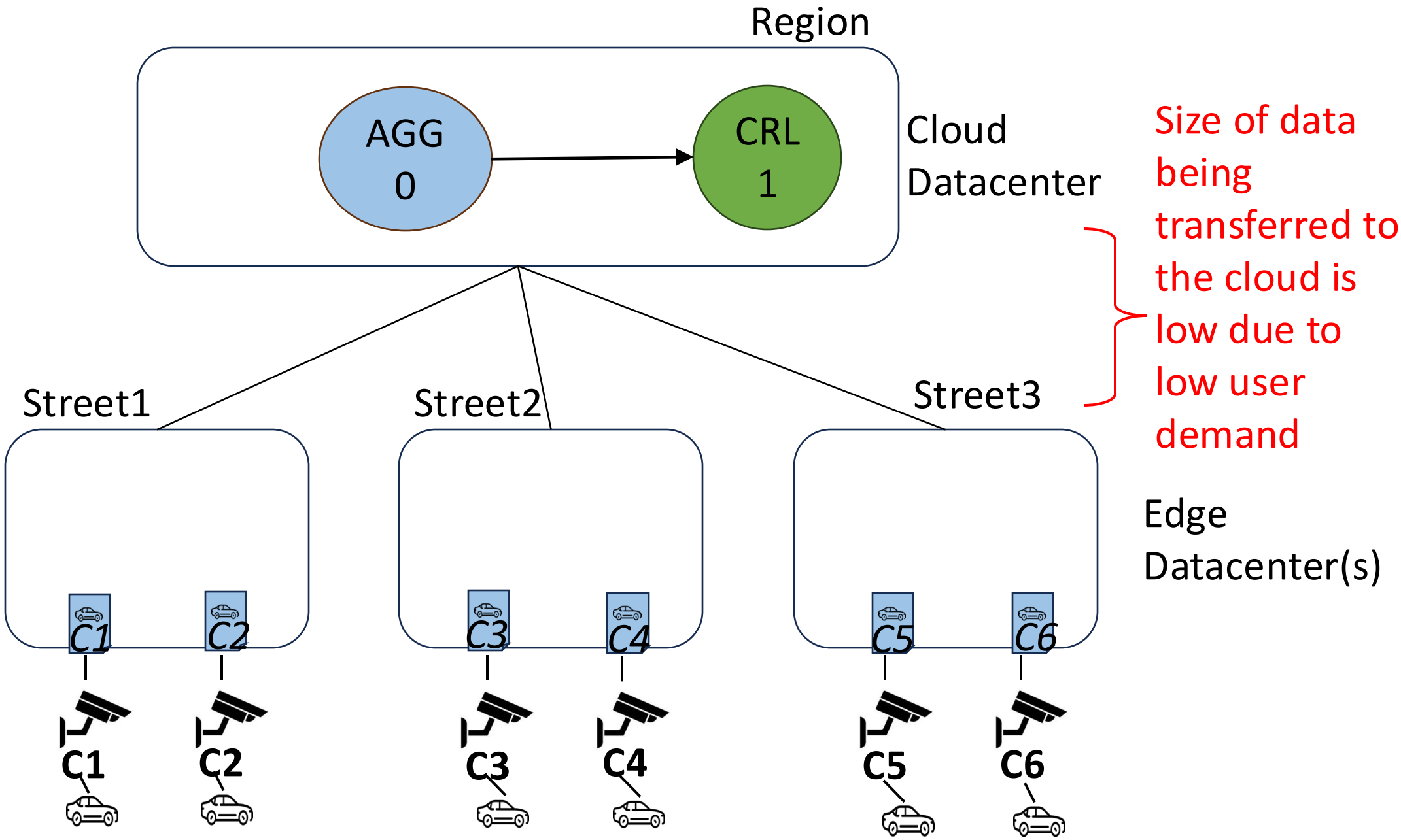
# Stateful Stream Processing - State

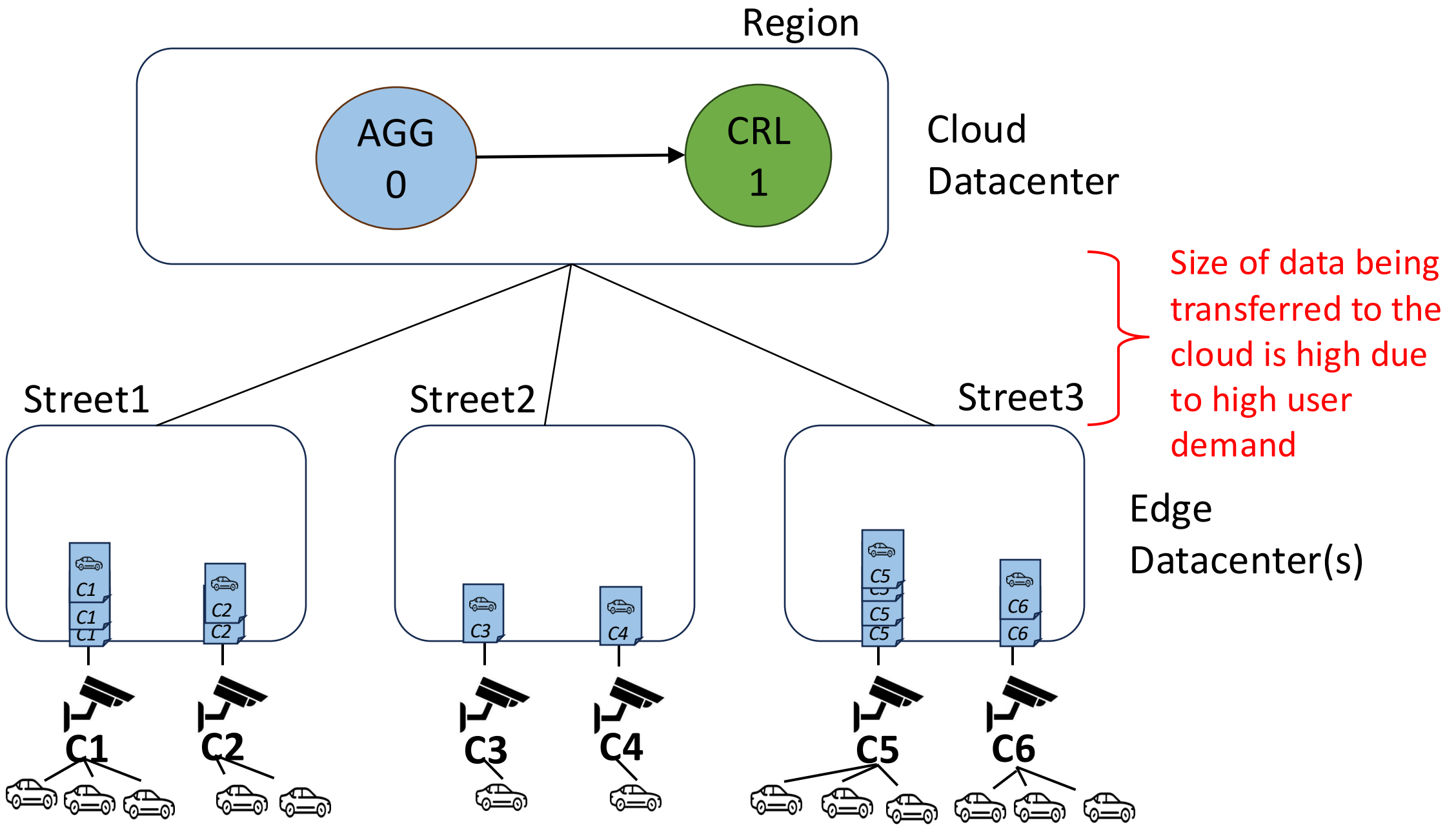


# Stateful Stream Processing - Windows

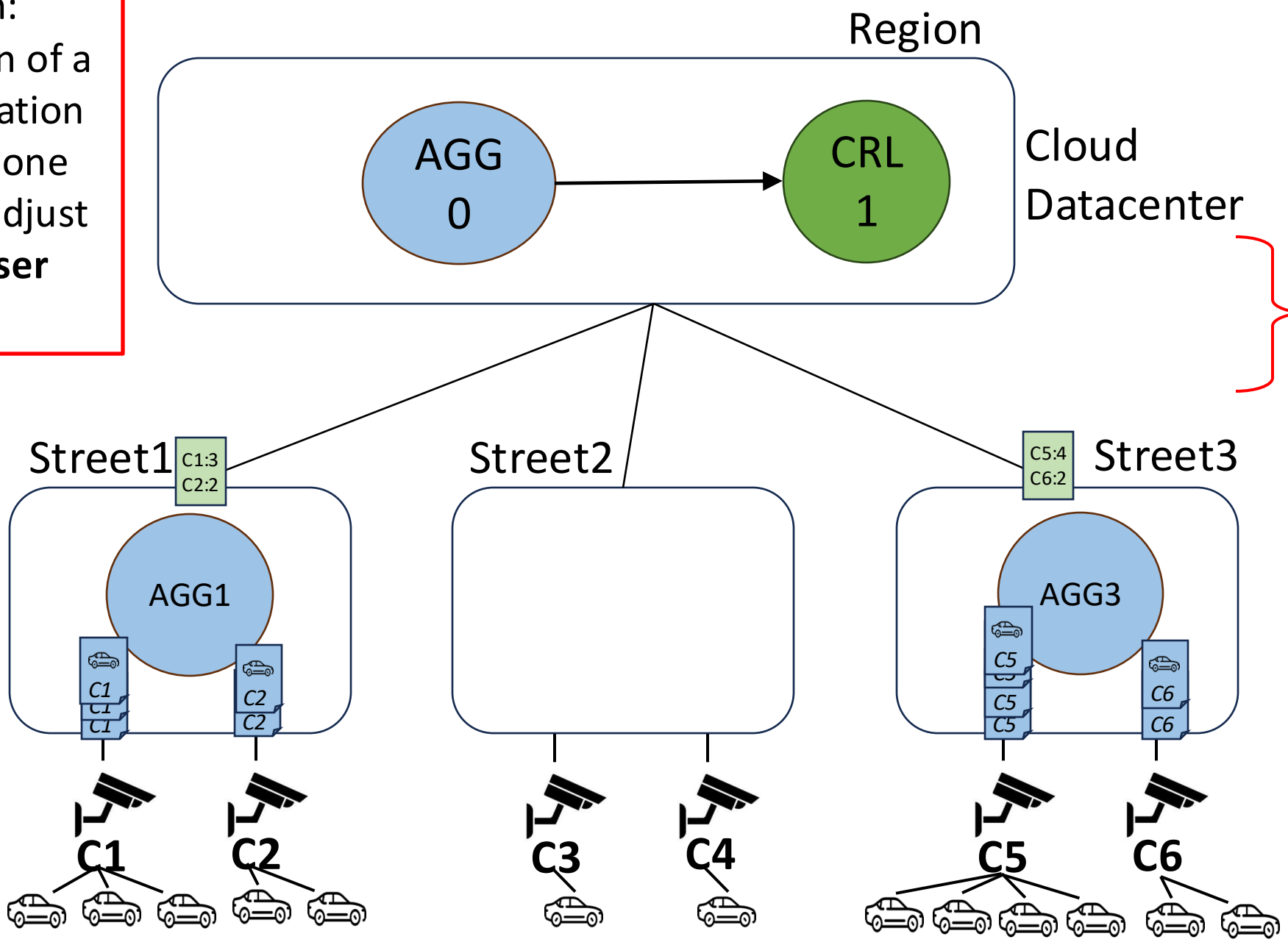


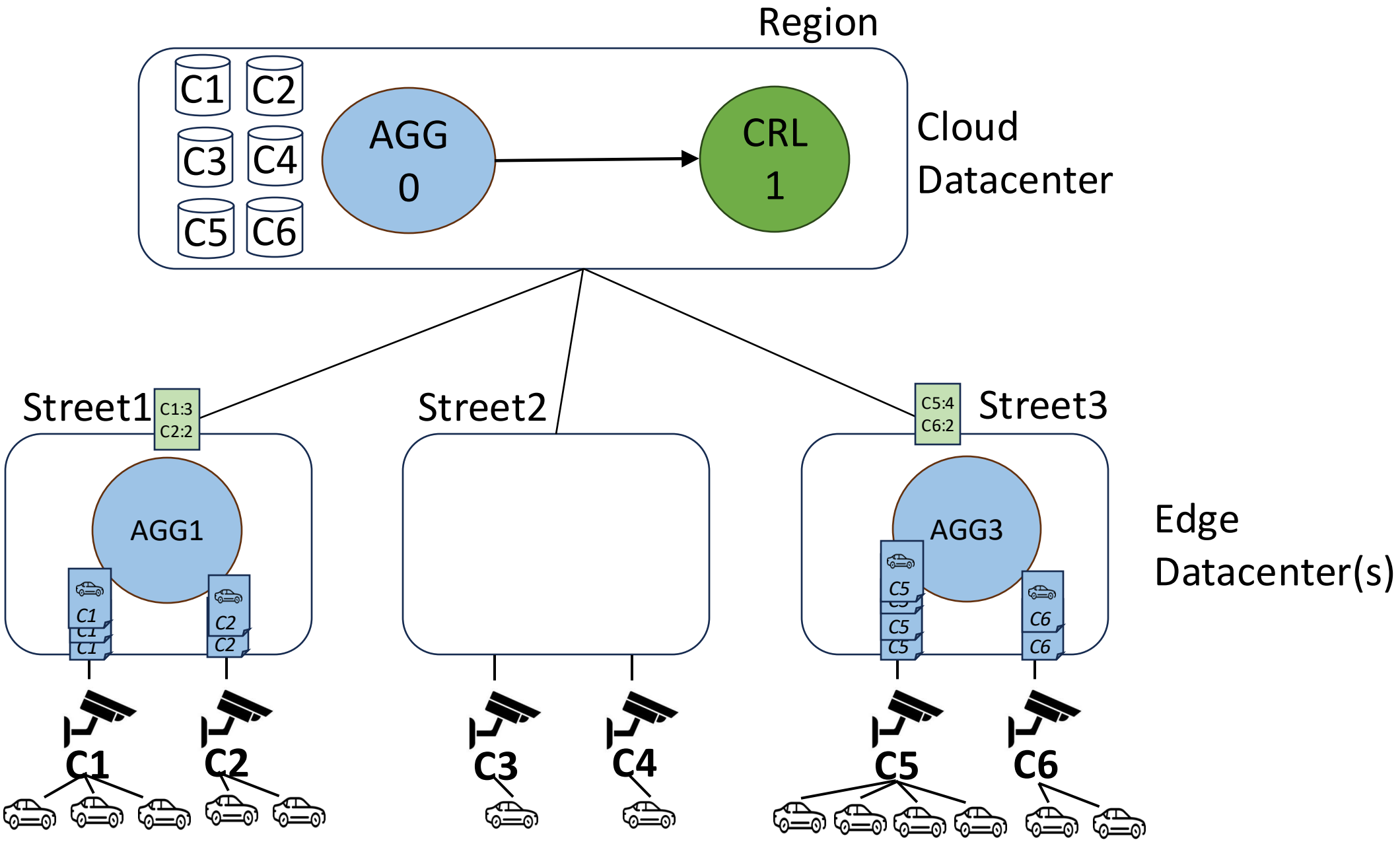

Region
AGG
0
CRL
1
Cloud
Datacenter
Latency and
bottlenecks
Street 1
Street 2
Street 3
Internet
C1
C2
C3
C4
C5
C6
C1
C2
C3
C4
C5
C6

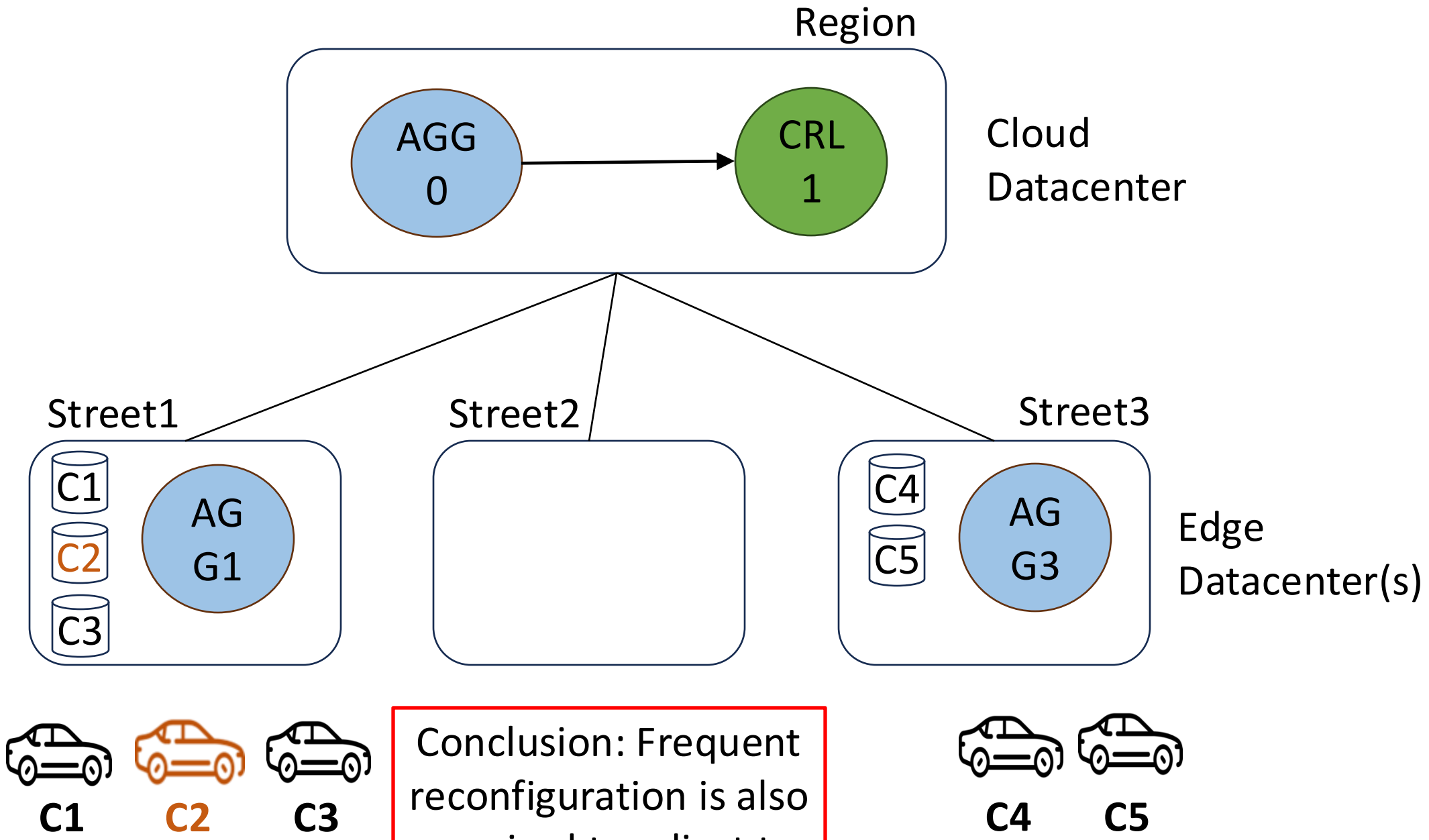

Region
AGG
0
CRL
1
Cloud
Datacenter
Size of data being transferred to the cloud is low due to low user demand
Street1
Street2
Street3
C1
C2
C3
C4
C5
C6
Edge
Datacenter(s)
C1
C2
C3
C4
C5
C6


Region
AGG
0
CRL
1
Cloud
Datacenter
Size of data being transferred to the cloud is high due to high user demand
Street1
Street2
Street3
Edge
Datacenter(s)
C1
C1
C1
C2
C2
C3
C4
C5
C5
C5
C6
C6
C1
C2
C3
C4
C5
C6

Conclusion: Reconfiguration of a running application needs to be done frequently to adjust to **varying user demand**.




Region
C1
C2
C3
C4
C5
C6
AGG
0
CRL
1
Cloud
Datacenter
Street1
C1:3
C2:2
Street2
C5:4
C6:2
Street3
AGG1
AGG3
C1
C2
C5
C6
C1
C2
C3
C4
C5
C6
Edge
Datacenter(s)

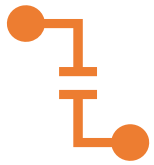



Conclusion: Frequent reconfiguration is also required to adjust to **moving data sources**.

# Need For Frequent Adaptation

- Edge resources are more expensive:
  - Use resources only when needed
- Data sources may move:
  - State may need to be moved between replicas

# Existing Systems



## Full-restart

Stops the entire application, performs reconfiguration and then restarts the application

Systems: **Flink**

## Partial-pause

Stops only the affected operators and Use fine-grained transfer and on-demand fetch to spread the impact of application disruption

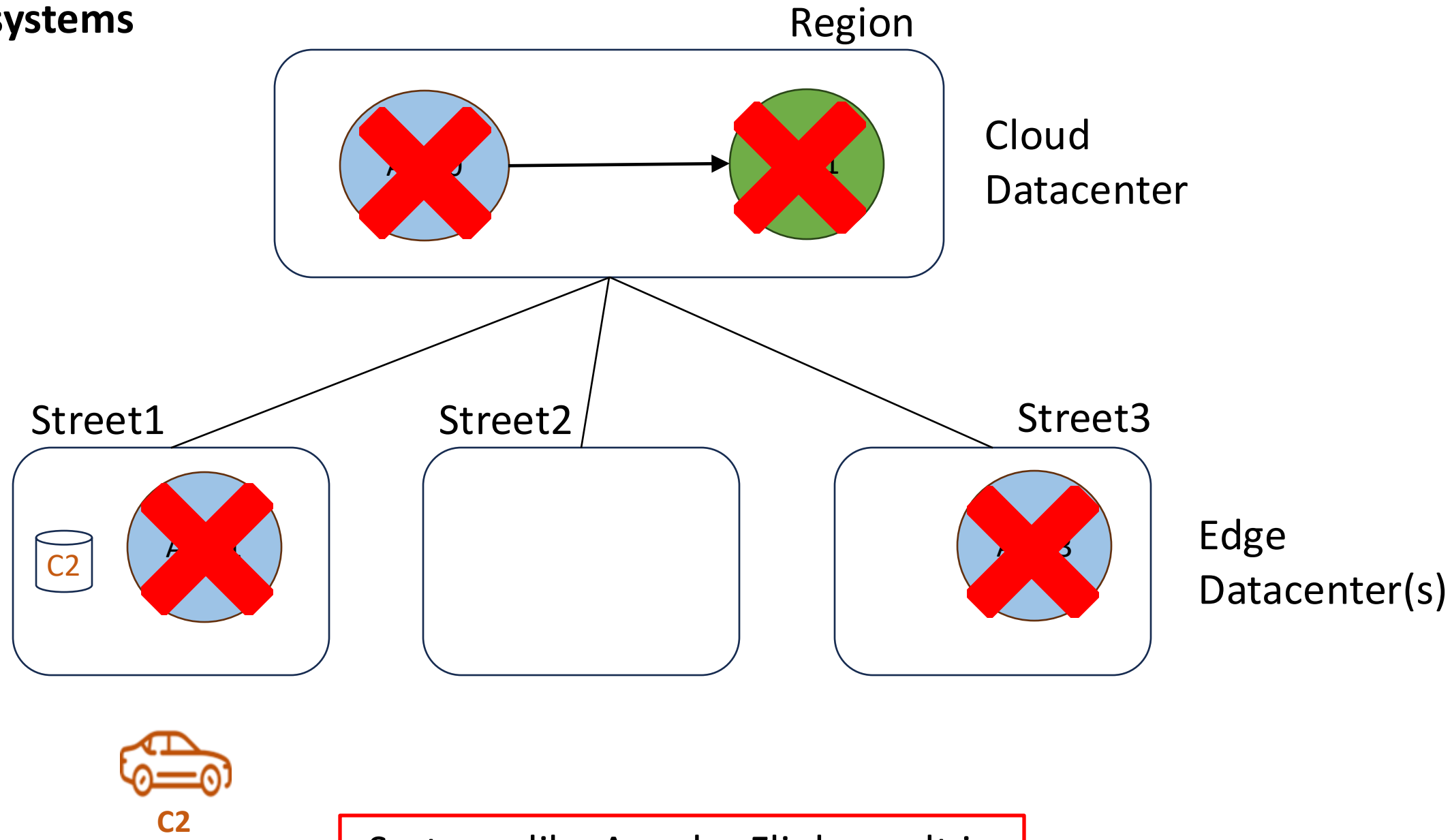
Systems: **Trisk**, **Meces**

## Hot backups

State is replicated periodically to all possible locations where reconfiguration could occur

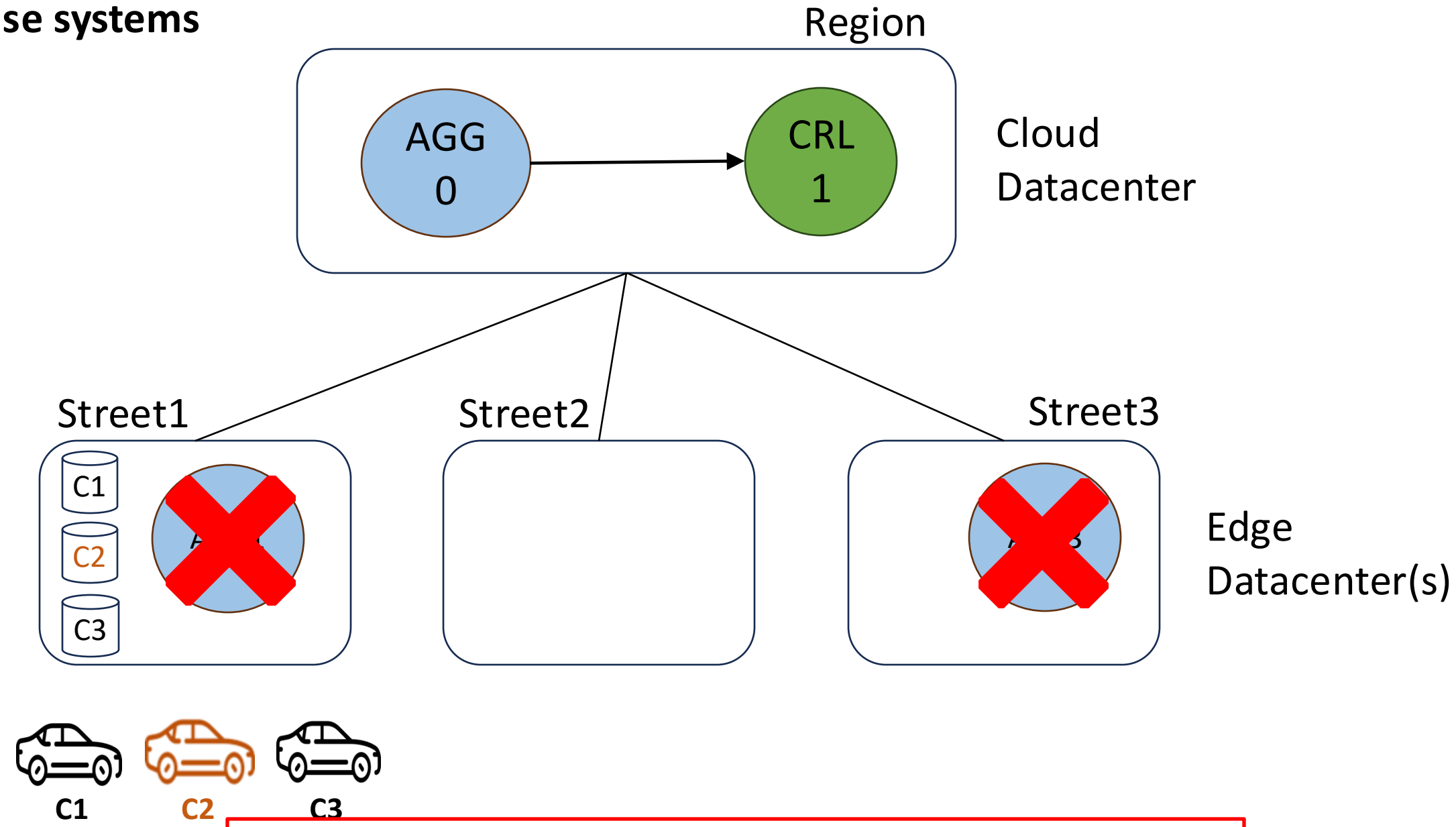
Systems: **Rhino**

# Full-restart systems



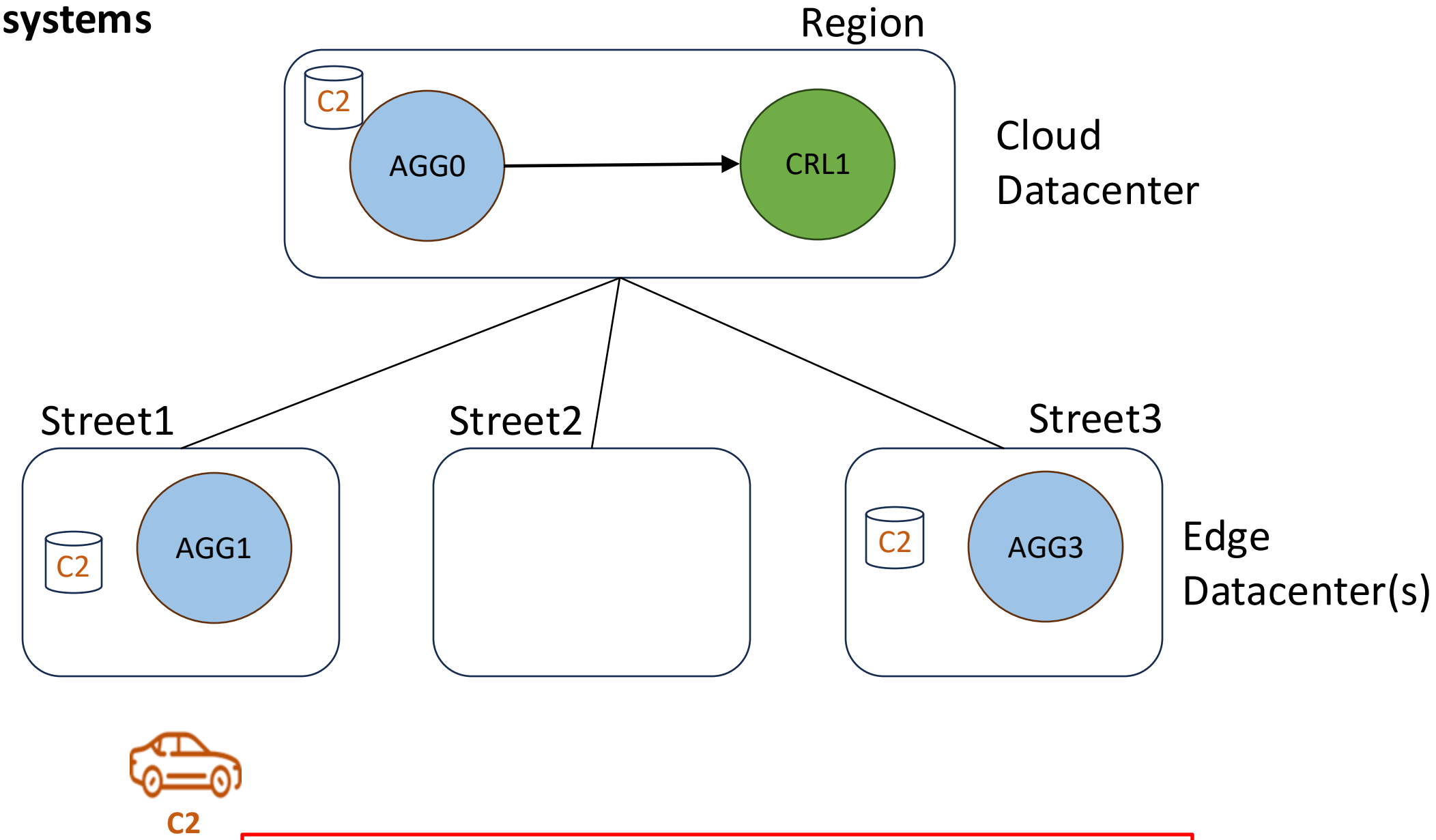

Systems like Apache Flink result in more delays

# Partial-pause systems




Systems like Trisk and Meces incurs disruption delays with inefficient state management.

# Hot backup systems




Hot backups systems like Rhino still incur disruption delays.

# Existing Solutions Fail For The Edge-Cloud!

- Reconfiguration has significant stoppage time.
  - Apache Flink
- Inefficient state management.
  - Meces
  - Rhino
  - Trisk
- Do not handle source mobility.

# Falcon: Live Reconfiguration for Stateful Stream Processing on the Edge

Pritish Mishra, Nelson Bore, Brian Ramprasad, Myles Thiessen,
Moshe Gabel, Alexandre da Silva Veith, Oana Balmau, Eyal de Lara



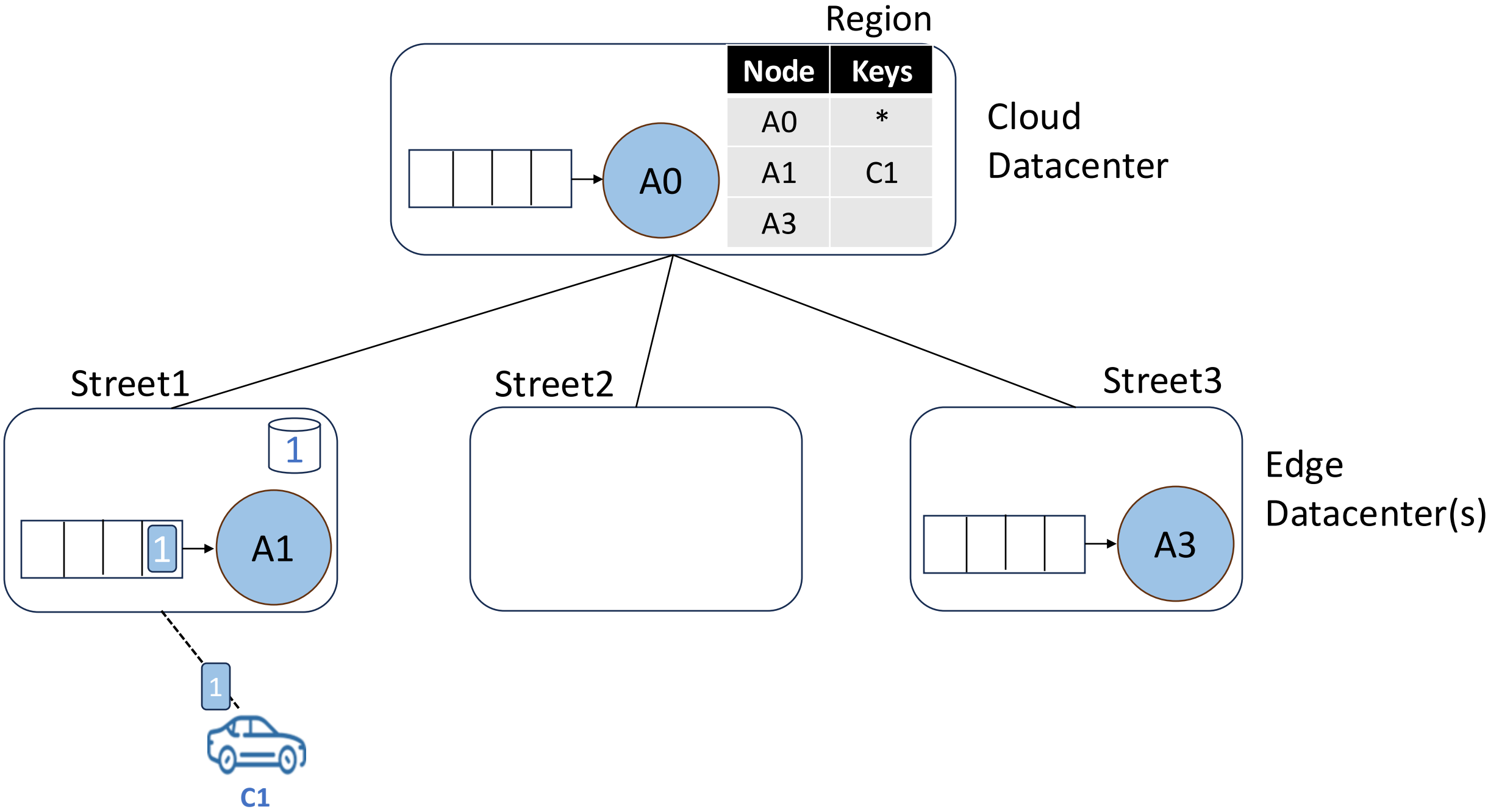
# Falcon (SEC'24)

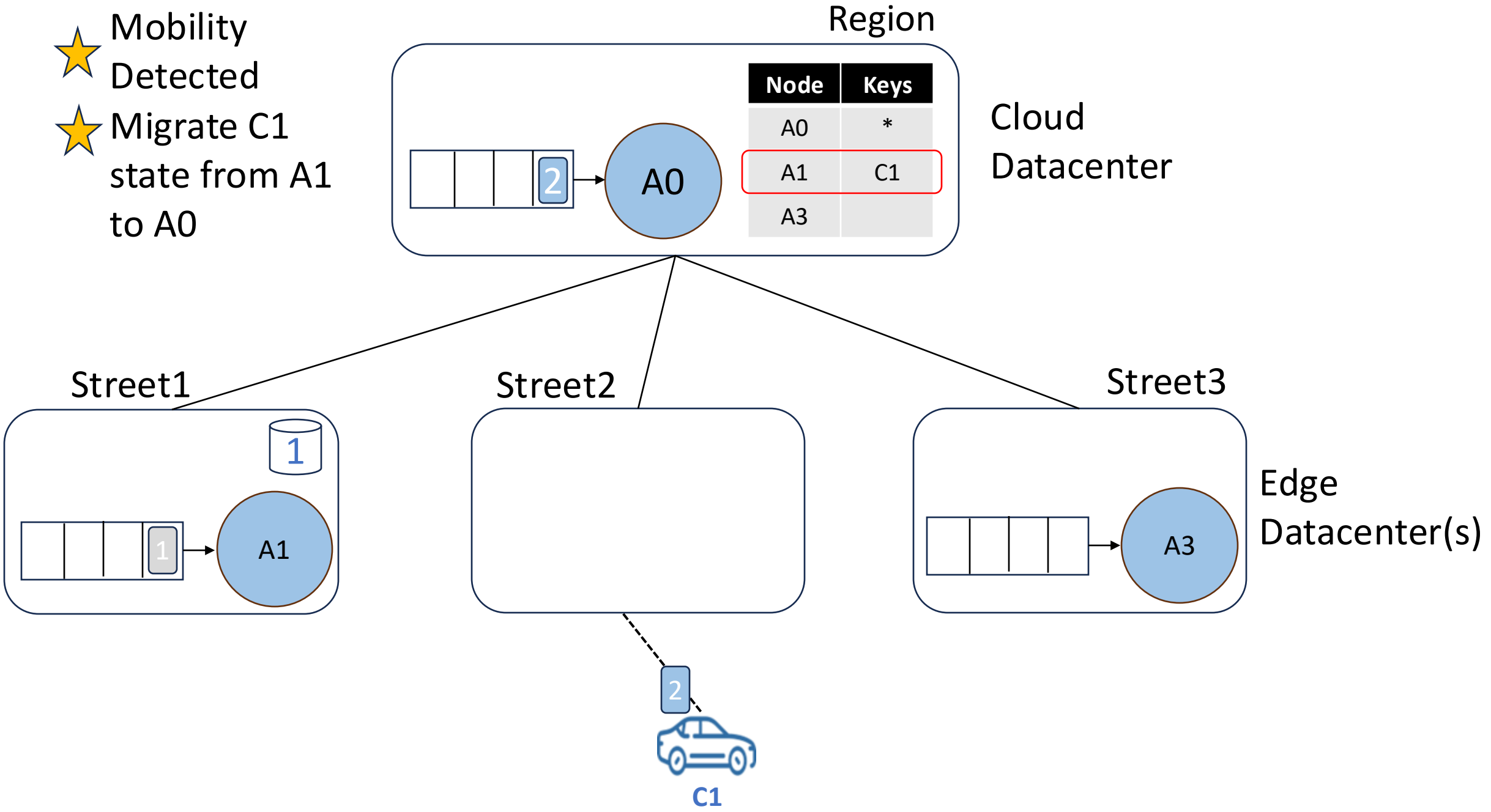
- Seamless reconfiguration of streaming operators
- Minimal stoppage time
- State and mobility support

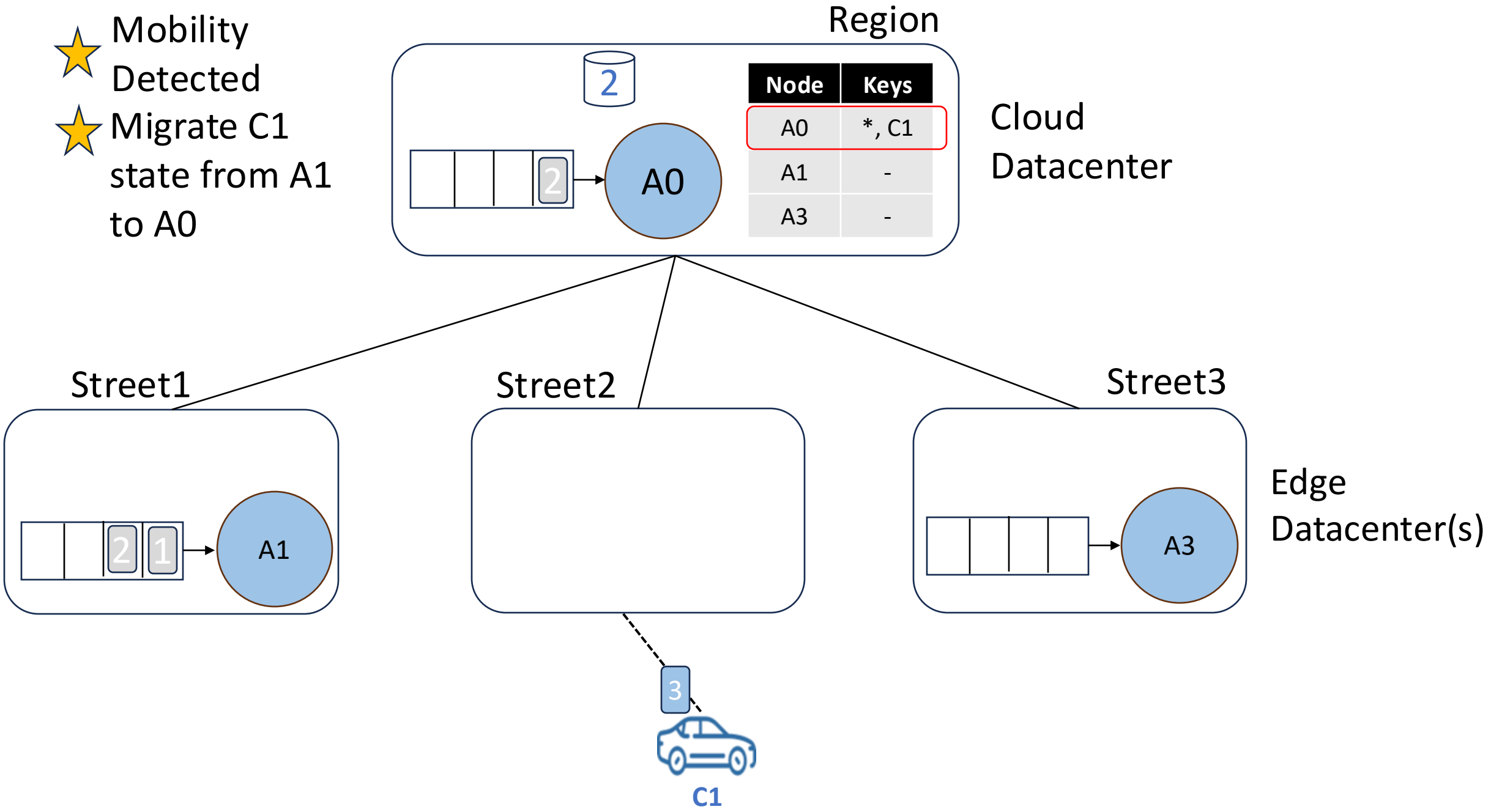
# Challenges Of Performing Reconfiguration

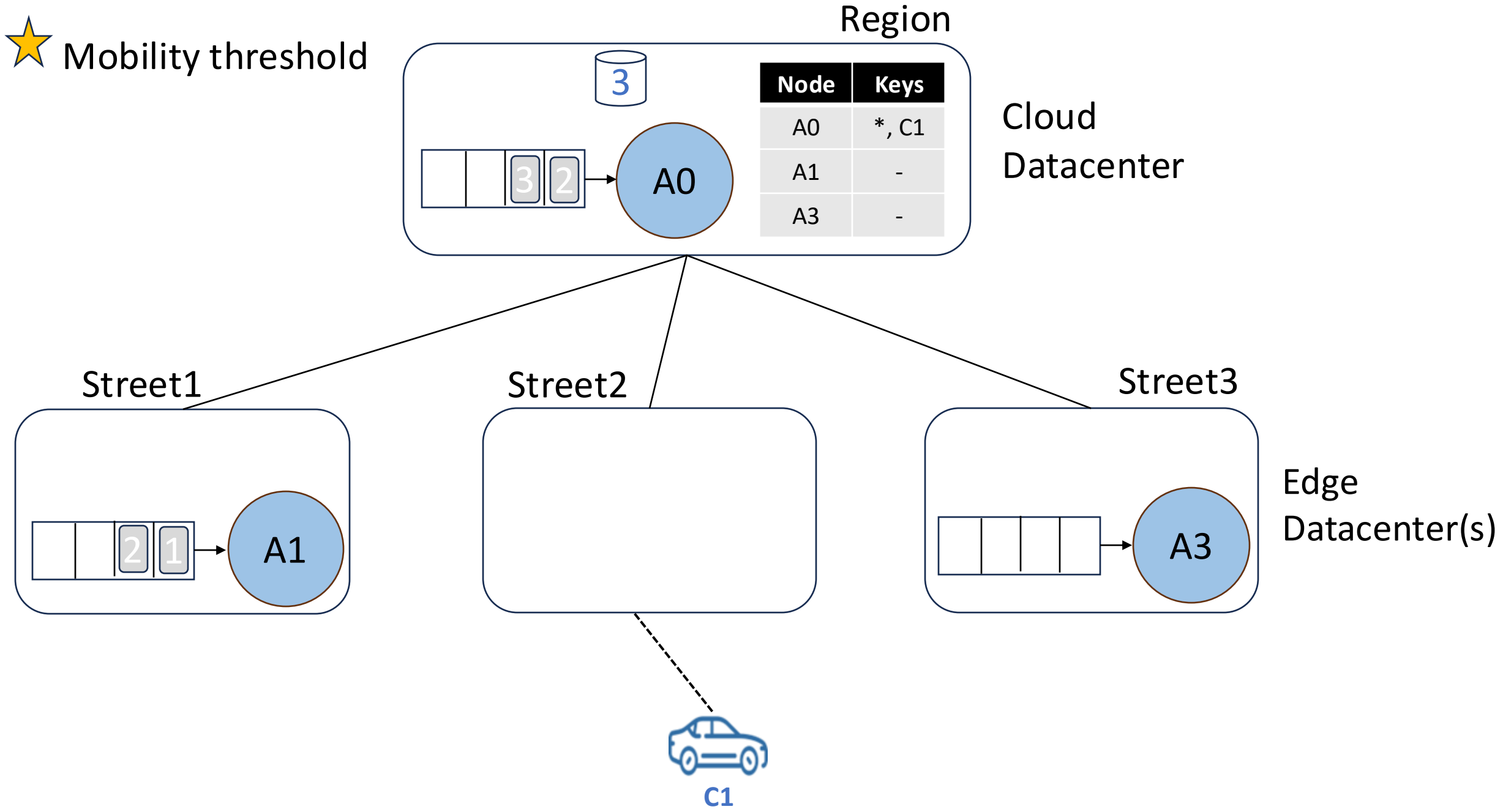
1. **Global coordination**
2. **Long state transfer time**
3. **State correctness guarantees**
4. **In-order and Exactly-once Tuple Processing**

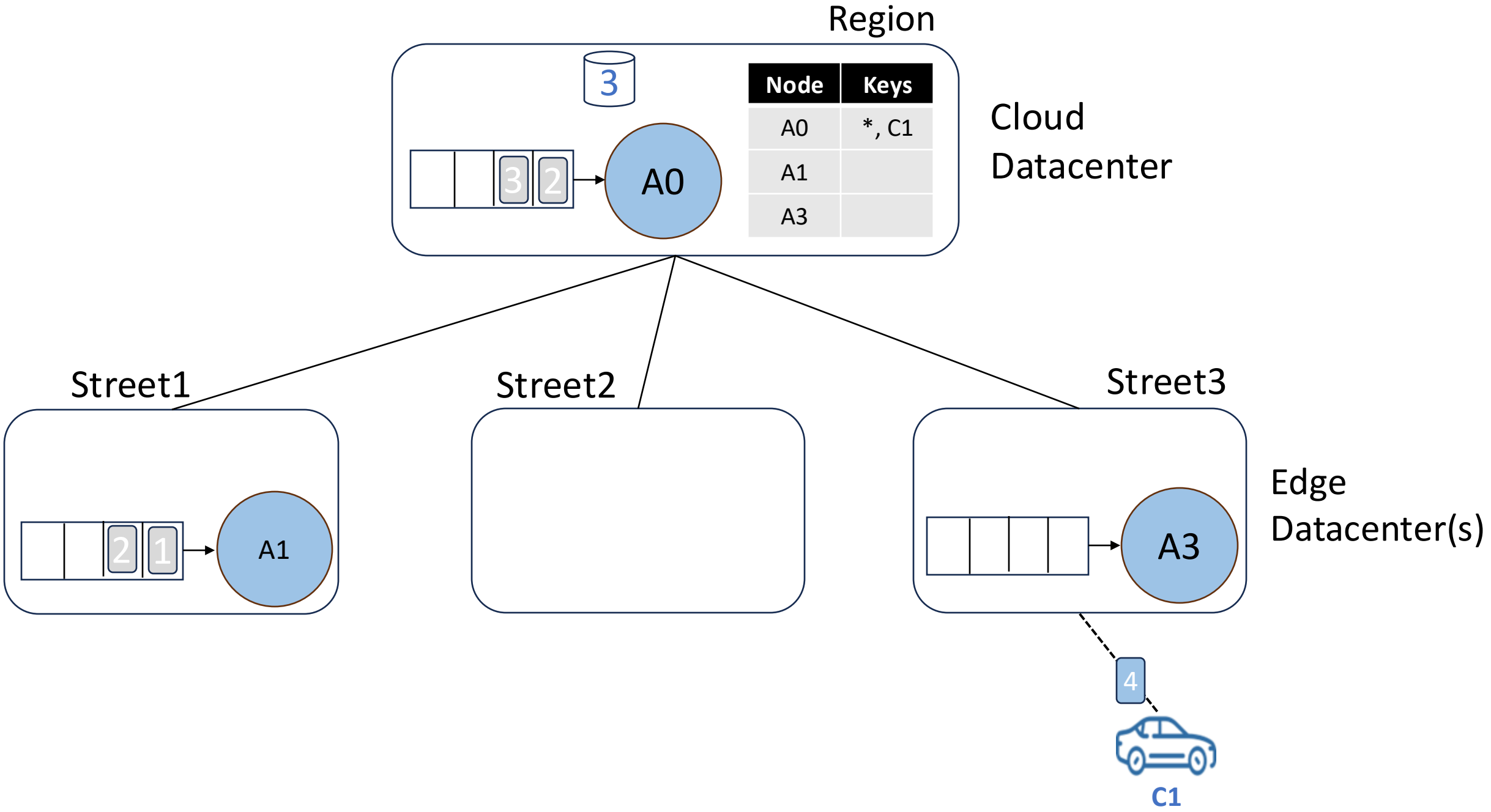
1. **Late Binding**
2. **Duplicate processing of tuples during reconfiguration**
3. **The source processes while the destination transfers and replays tuples to sync its state**
4. **Emit filter and Marker-based synchronization**

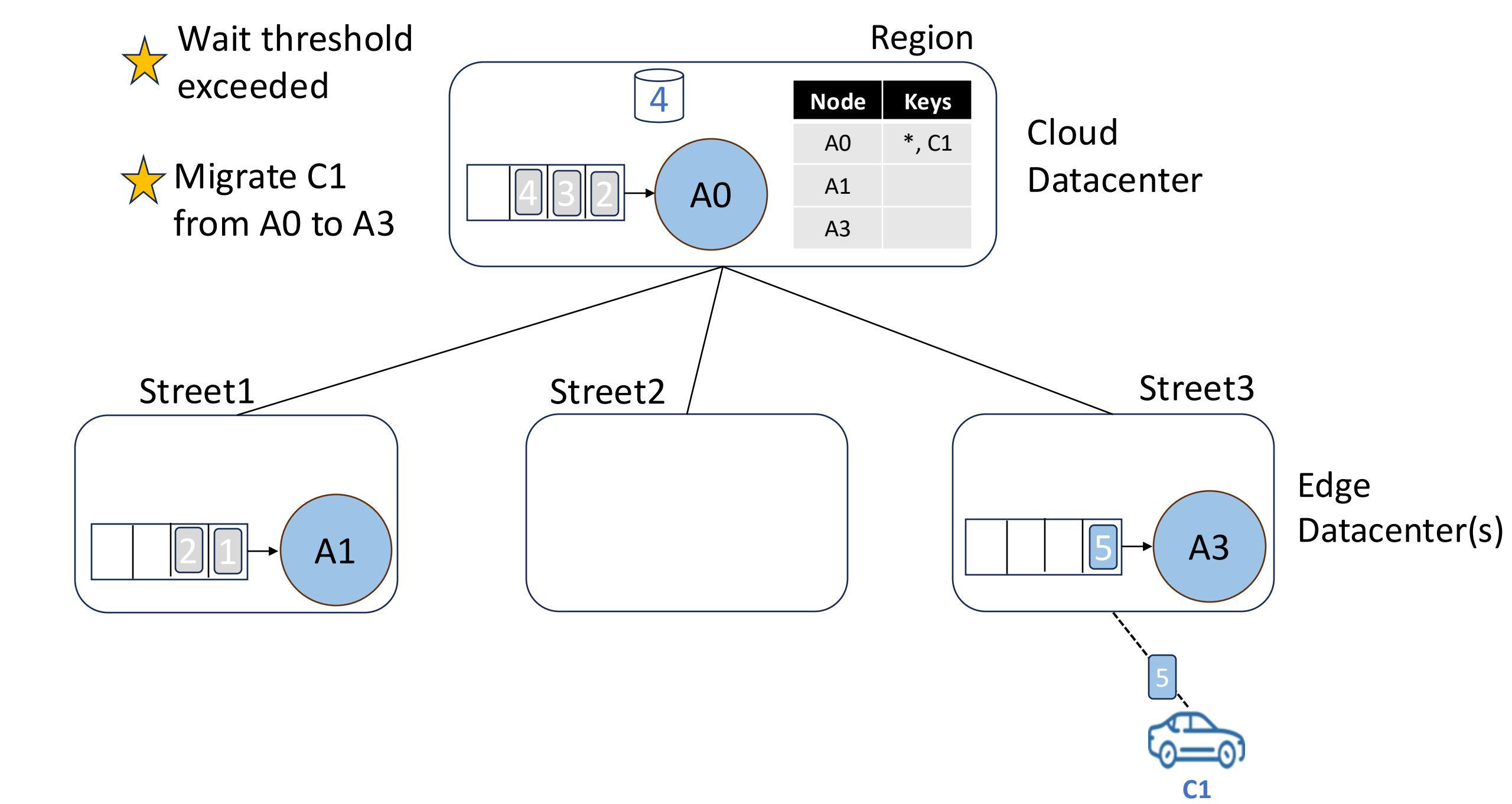

Region
Node
Keys
A0
*
A1
C1
A3
A0
Cloud
Datacenter
Street1
Street2
Street3
1
1
A1
A3
Edge
Datacenter(s)
1
C1

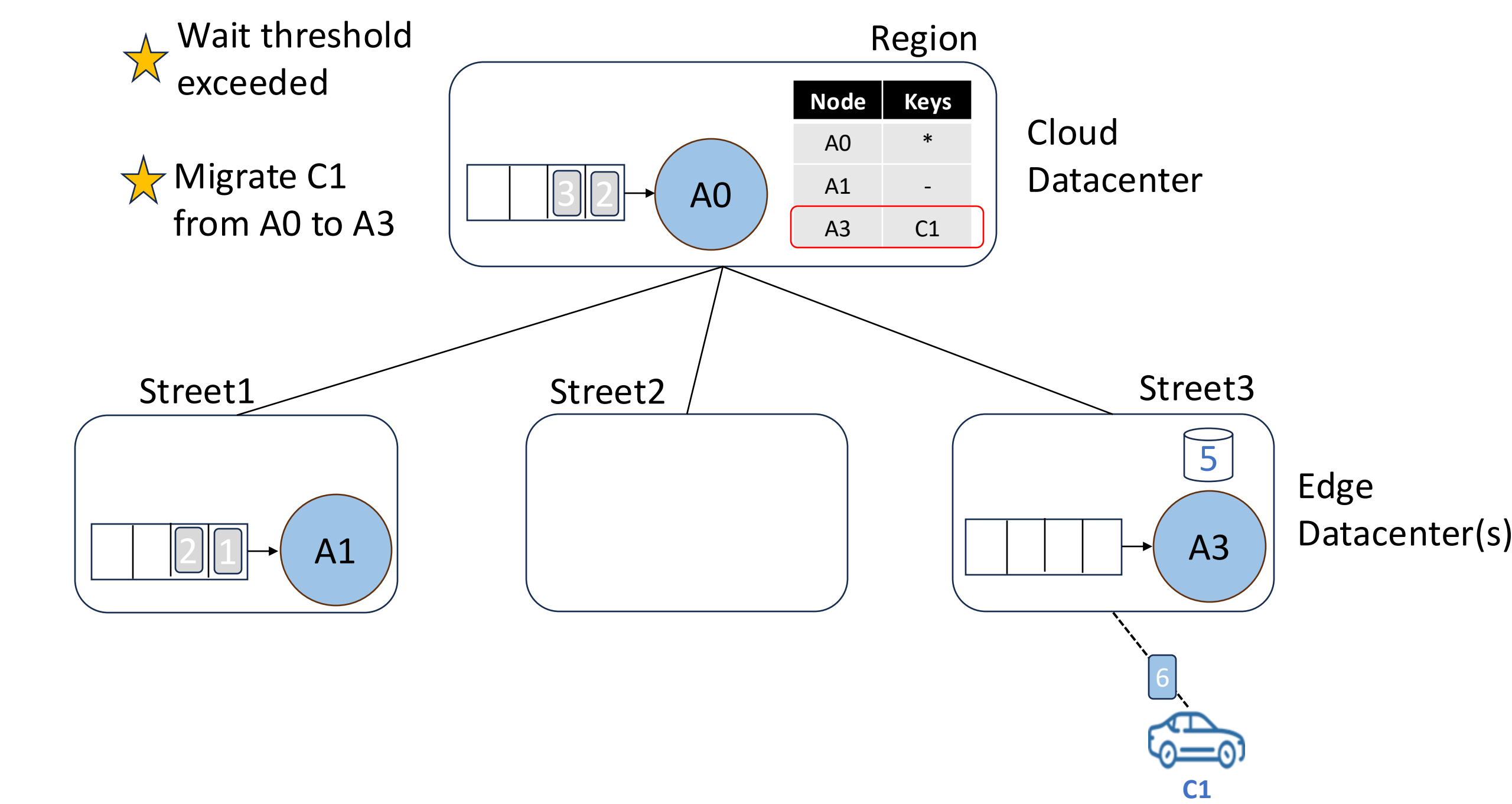

Mobility Detected
Migrate C1 state from A1 to A0
Region
Node
Keys
A0
*
A1
C1
A3
2
A0
Cloud Datacenter
Street1
Street2
Street3
1
1
A1
A3
Edge Datacenter(s)
2
C1

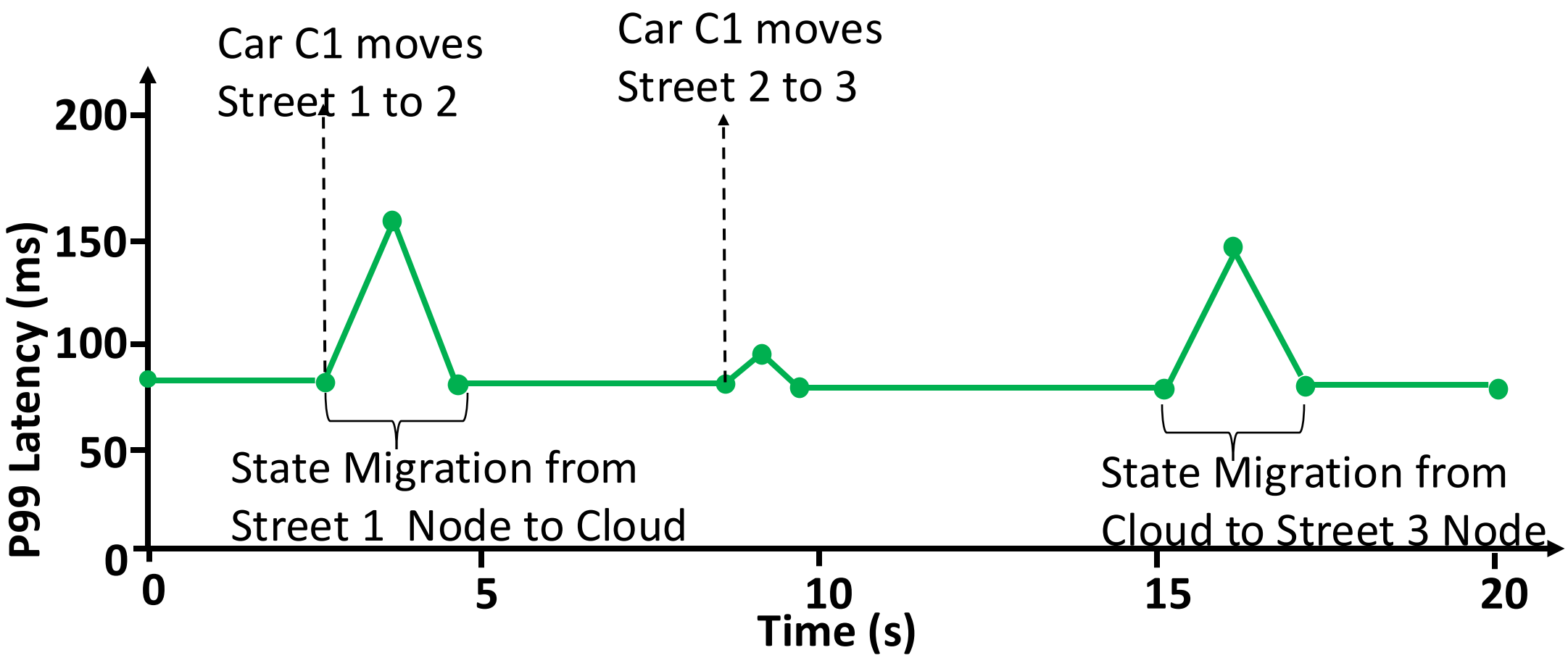

Mobility Detected
Migrate C1 state from A1 to A0
Region
2
Node
Keys
A0
*, C1
A1
-
A3
-
2
A0
Cloud Datacenter
Street1
Street2
Street3
2
1
A1
A3
Edge Datacenter(s)
3
C1


Mobility threshold
Region
3
Node
Keys
A0
*, C1
A1
-
A3
-
3
2
A0
Cloud
Datacenter
Street1
Street2
Street3
2
1
A1
A3
Edge
Datacenter(s)
C1


Region
3
Node
Keys
A0
*, C1
A1
A3
3
2
A0
Cloud
Datacenter
Street1
Street2
Street3
2
1
A1
A3
Edge
Datacenter(s)
4
C1

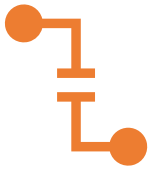

Wait threshold exceeded
Migrate C1 from A0 to A3
Region
4
Node
Keys
A0
*, C1
A1
A3
4 3 2
A0
Cloud Datacenter
Street1
Street2
Street3
2 1
A1
5
A3
Edge Datacenter(s)
5
C1


Wait threshold exceeded
Migrate C1 from A0 to A3
Region
Node
Keys
A0
*
A1
-
A3
C1
3
2
A0
Cloud Datacenter
Street1
Street2
Street3
2
1
A1
5
A3
Edge Datacenter(s)
6
C1




**Car (C1) moves from Street 1 Edge Node to Street 3 Edge Node**

# Evaluation Baselines



## Full-restart

Stops the entire application, performs reconfiguration and then restarts the application

Systems: **Flink**

## Partial-pause

Stops only the affected operators and Use fine-grained transfer and on-demand fetch to spread the impact of application disruption

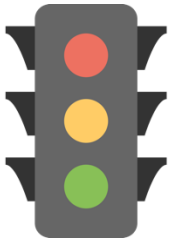
Systems: **Trisk**, **Meces**

## Hot backups

State is replicated periodically to all possible locations where reconfiguration could occur

Systems: **Rhino (Falcon-HB)**

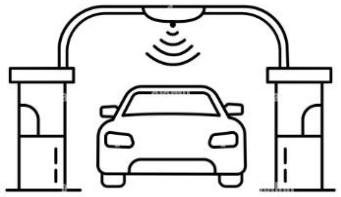
# Benchmark Apps



## Traffic Monitoring

Detects speeding vehicles using a running average

Allows for easy control of number of keys, state size



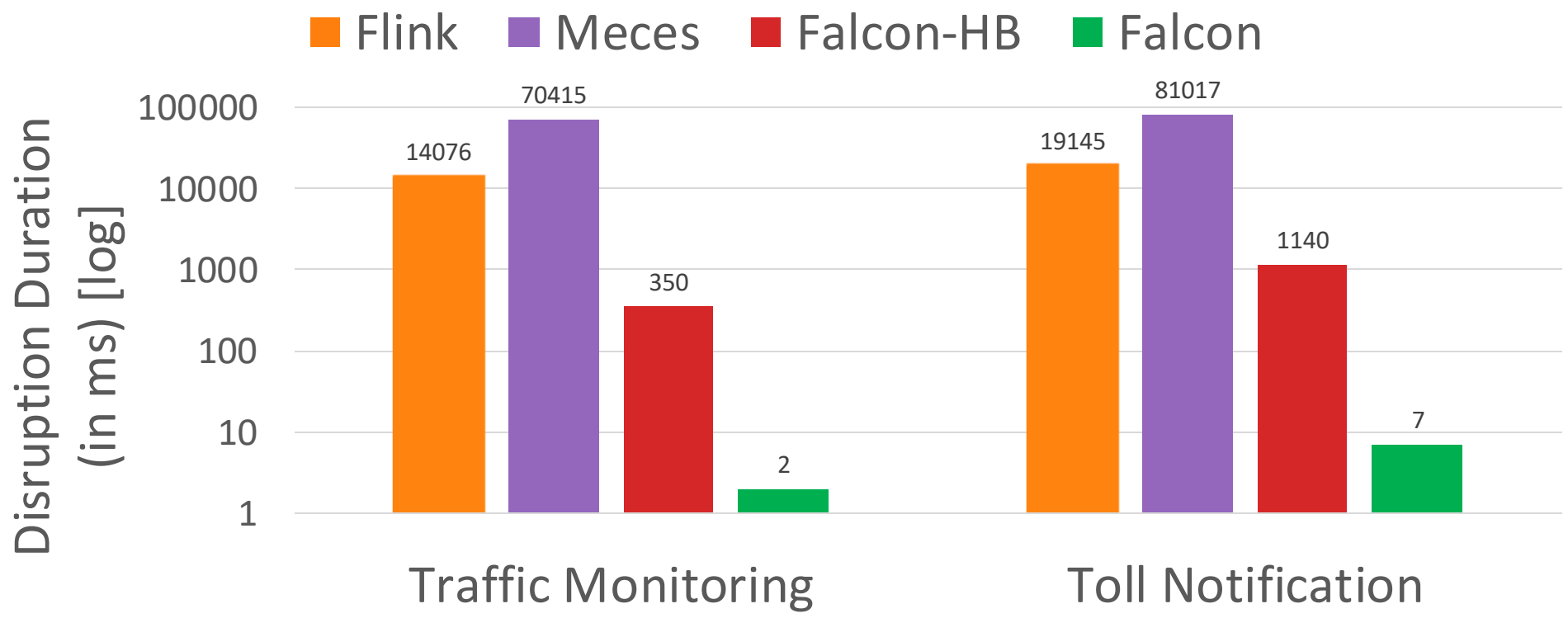
## Toll Notification

Smart toll computation and real-time accident detection

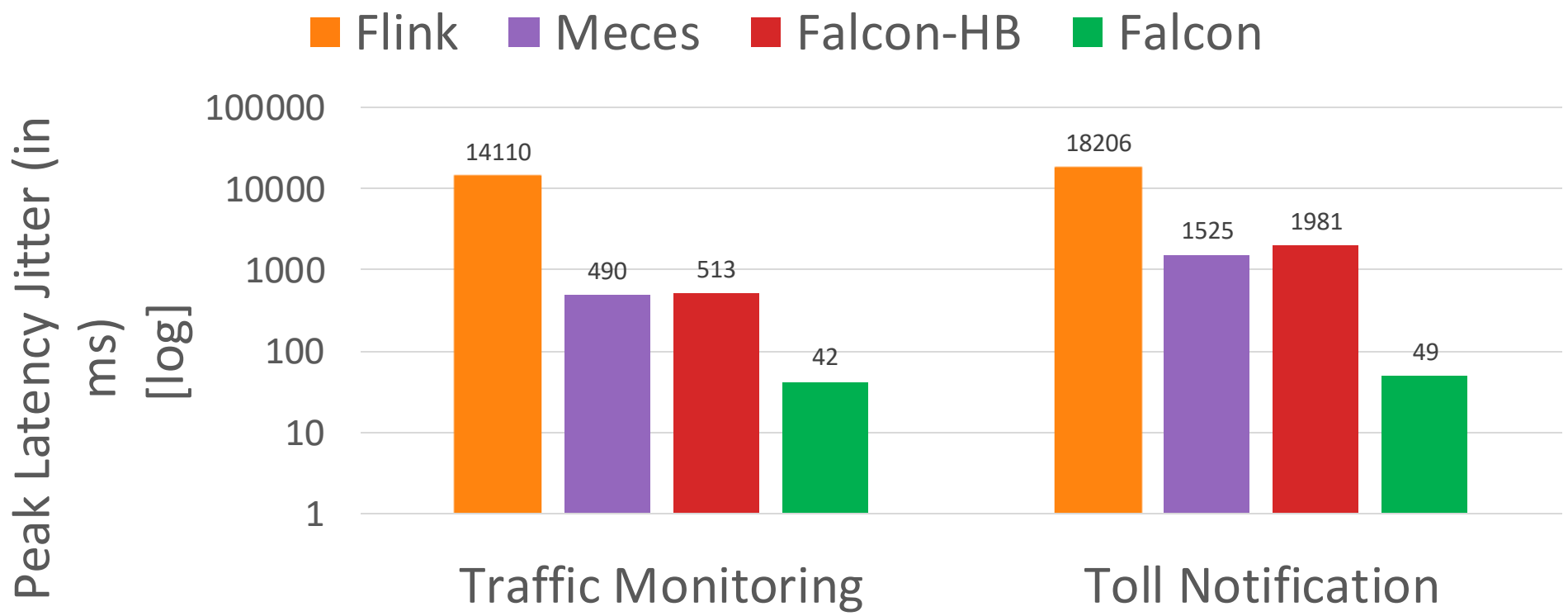
# Road Segments - 101
# Cars (Keys) – 124K

Uses event and count-based sliding windows

# Disruption Duration



# Peak Latency Jitter



# Summary

**Data management on the edge-cloud is a difficult problem:**

- Existing approaches do not support seamless reconfiguration
- Global coordination, Early binding

**Falcon:**

- Uses hierarchical network of routers
- Supports state transfer and data source mobility

**Thanks to students and collaborators!!**

*Eyal de Lara*
*Prof. Univ of Toronto*

*Pritish Mishra*
*PhD candidate*

*Nelson Bore*
*PhD candidate*

**Check out the full paper here:**

https://github.com/delara/falcon

Contact for Falcon: pritish@cs.toronto.edu

*discslab.cs.mcgill.ca*
*oana.balmau@mcgill.ca*