CellIQ: Real-Time Cellular Network Analytics at Scale

Anand Iyer[#], Li Erran Li[+], Ion Stoica[#]

[#]UC Berkeley [+]Bell Labs

Cellular Networks have been seeing **exponential** growth and become part of our lives

Image courtesy: Alcatel-Lucent

What is needed to solve these issues?

Are some regions in the network hotspots?

- Better load balancing

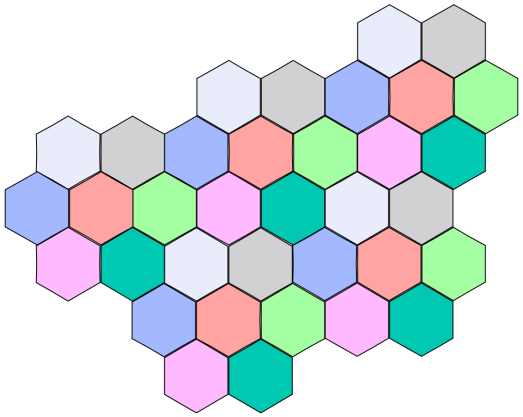
How is user traffic moving in the network?

- Better resource provisioning

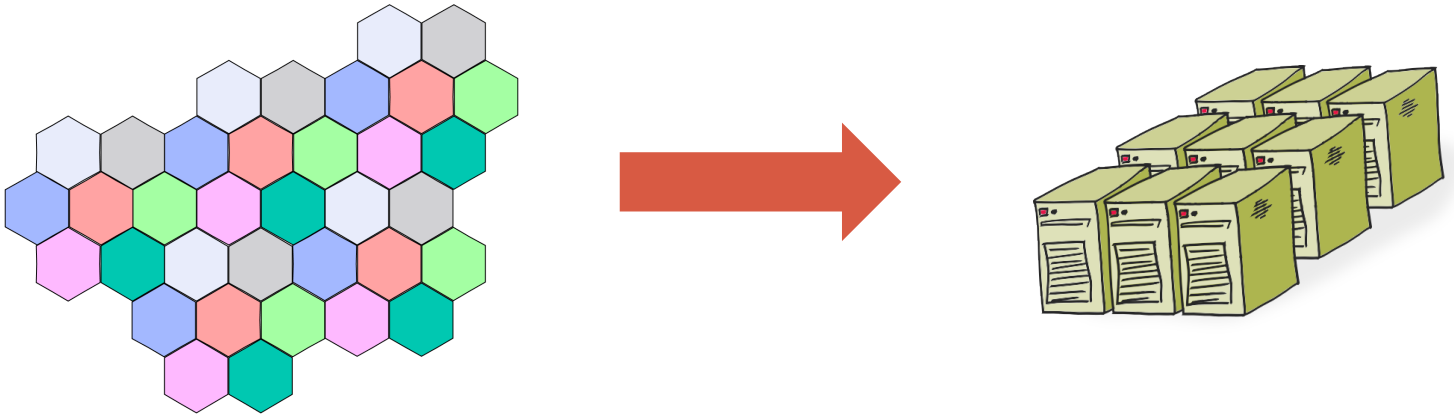
What are the popular handoff sequences?

- Troubleshoot handoff related problems

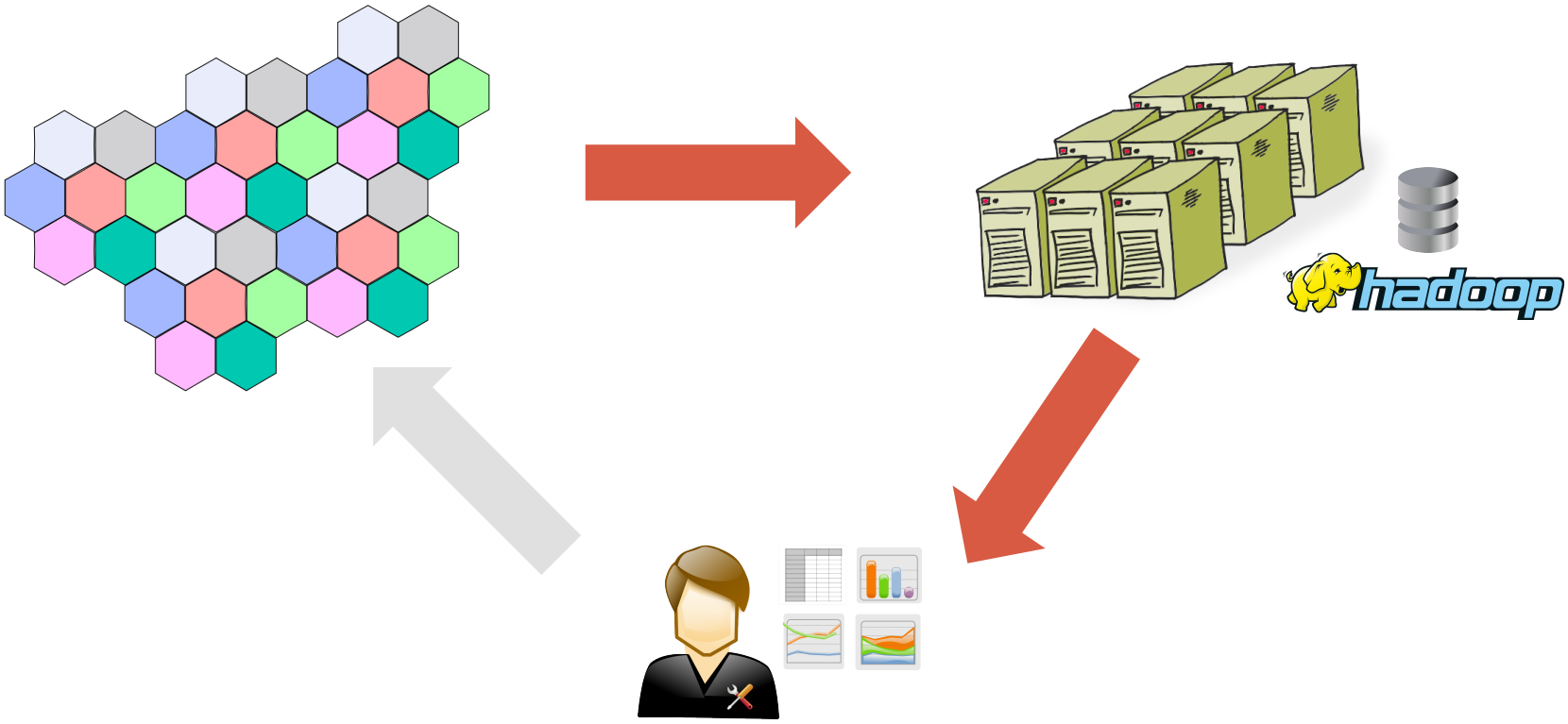
Cellular Network Analytics Today



Cellular Network Analytics Today



Cellular Network Analytics Today



Problem

Existing cellular network analytic systems do not support **advanced** analytic tasks in an **efficient** manner.

Challenges

High Velocity Data

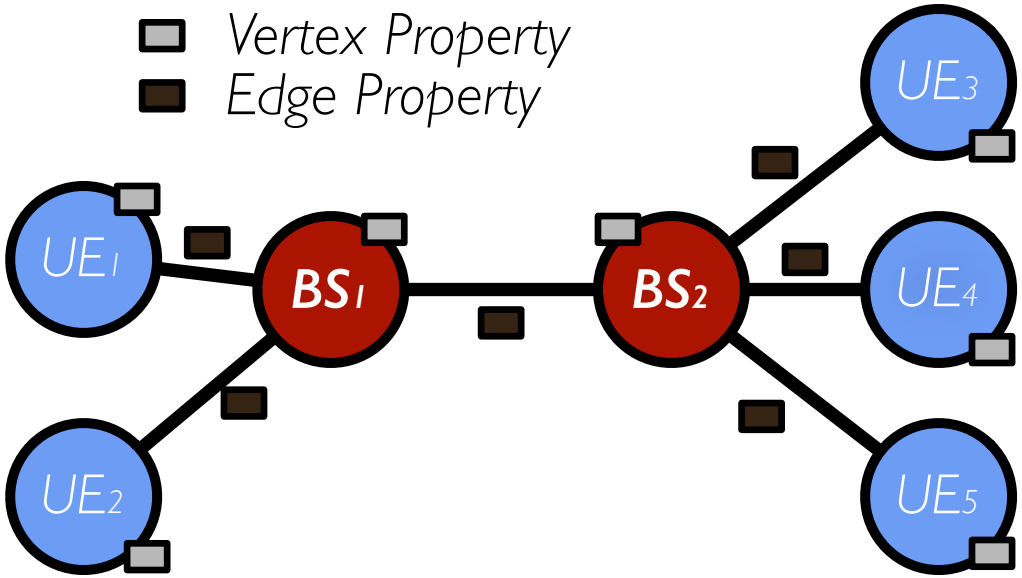
Continuous Monitoring

Advanced Tasks

Timely Spatio-Temporal Analysis

CellIQ is a cellular network analytics system that supports **rich** analysis tasks **efficiently** by leveraging domain-specific optimizations

Cellular Data as Time-Evolving Graphs

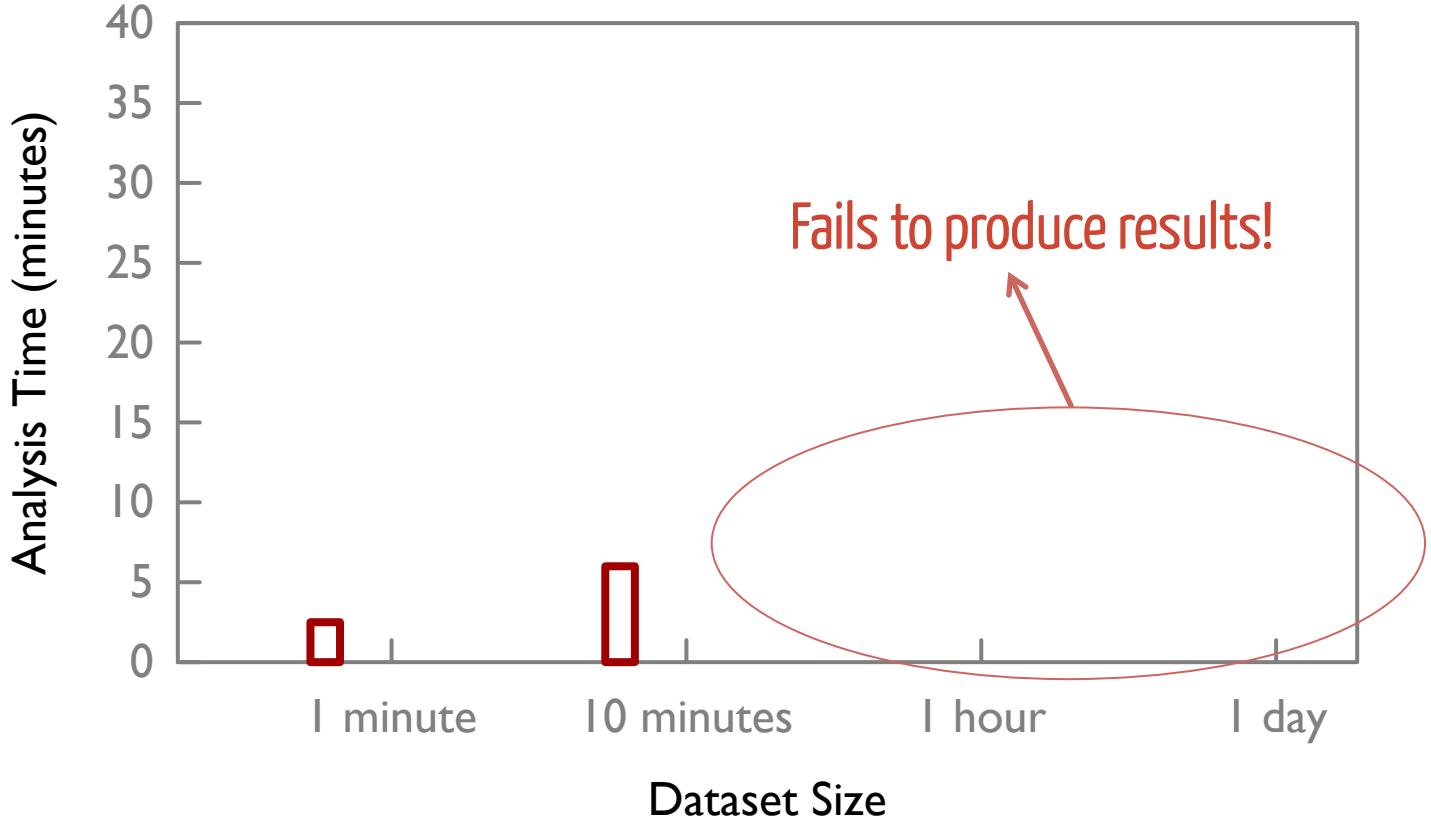


Tasks easily expressed in graphs:

Hotspot computation ➔ Connected components

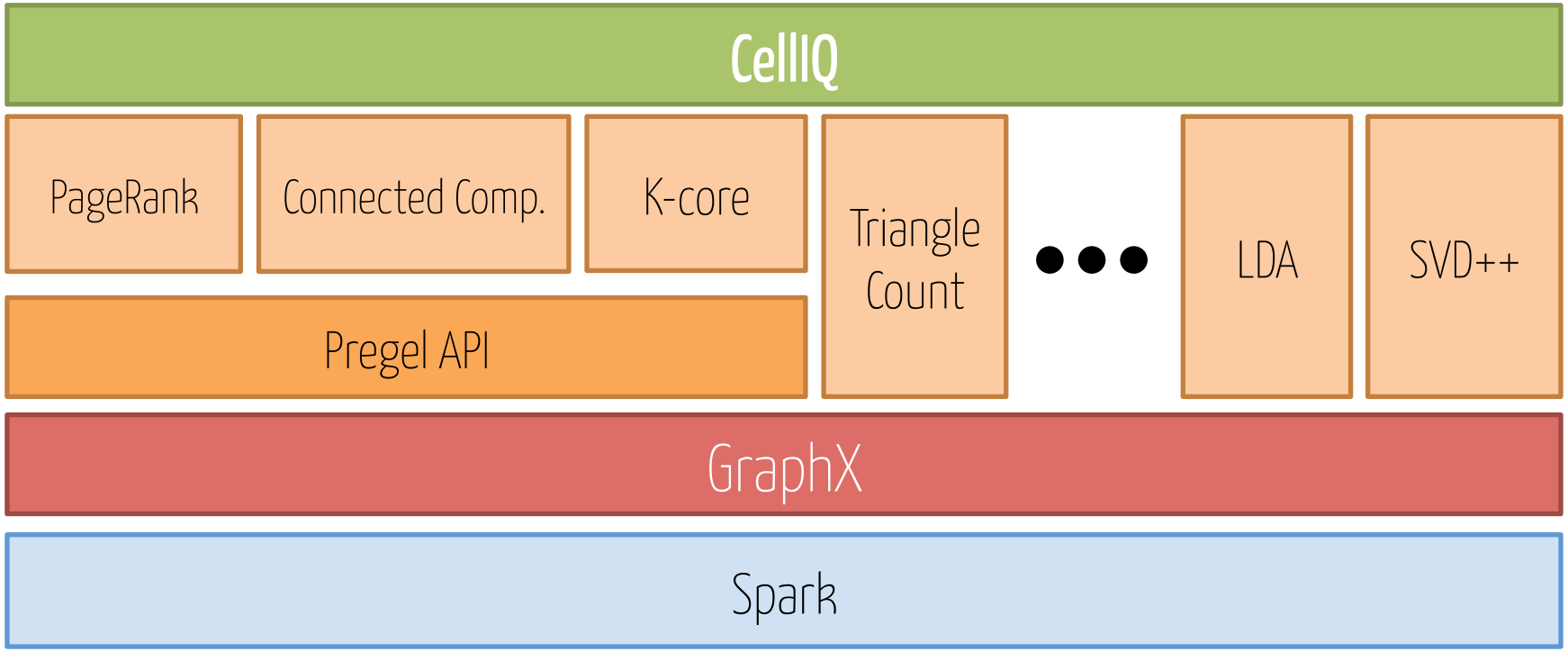
Handoff sequences & User traffic ➔ Pregel model

Why Not Use a Graph Parallel Framework?



Domain specific optimizations key for efficient analysis

CellIQ Implementation

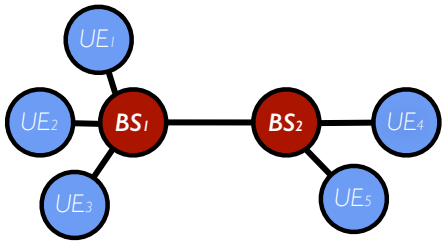


Implemented as a layer on GraphX*

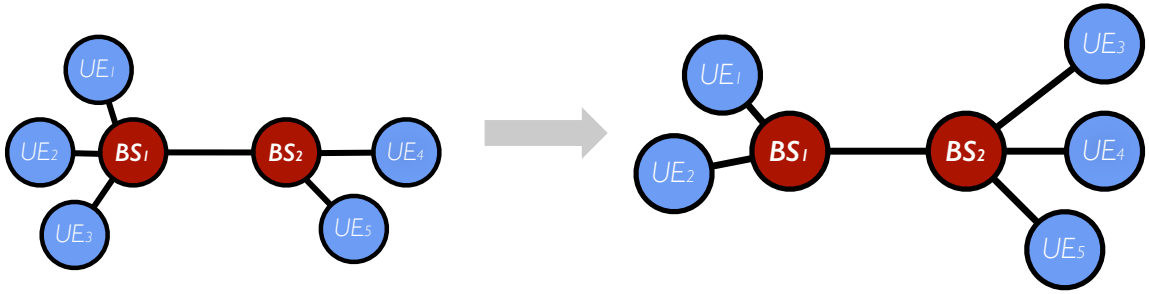
Incorporates several domain specific optimizations

*Gonzales. et.al. "GraphX: Graph Processing in a Distributed Dataflow Framework", OSDI 2014

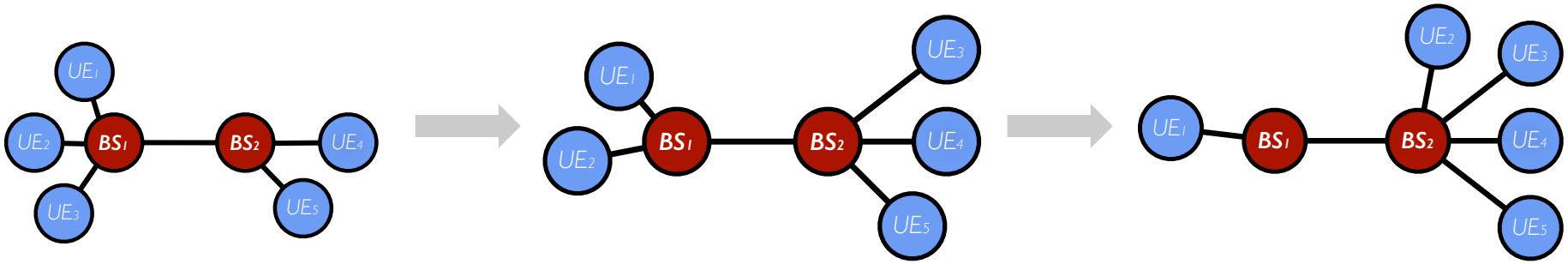
Computational Model



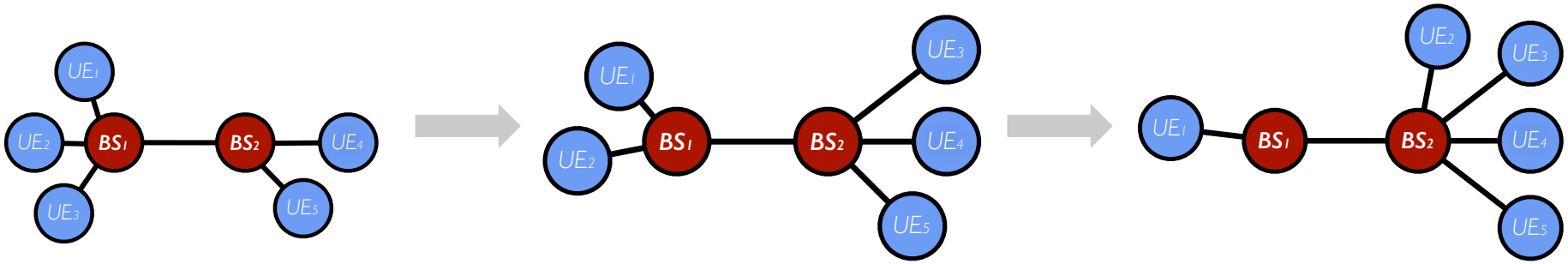
Computational Model



Computational Model



Computational Model: GStreams

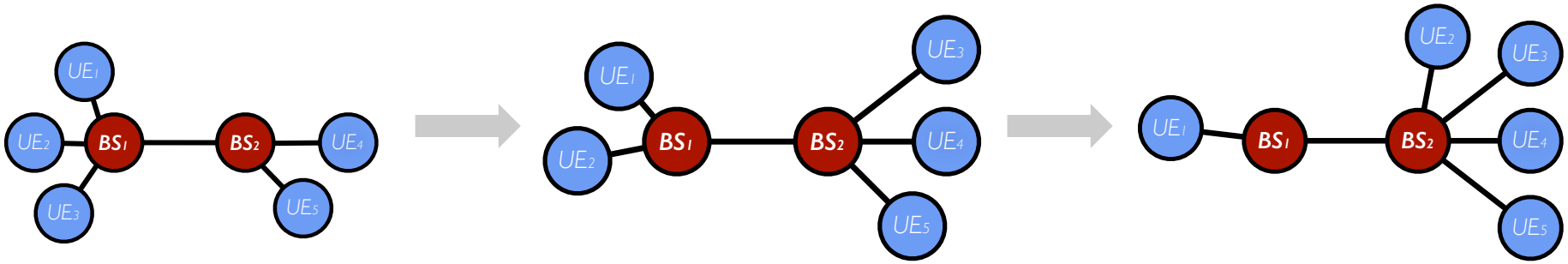


Domain specific graph partitioning

Spatial operations

Window operations

Computational Model: GStreams



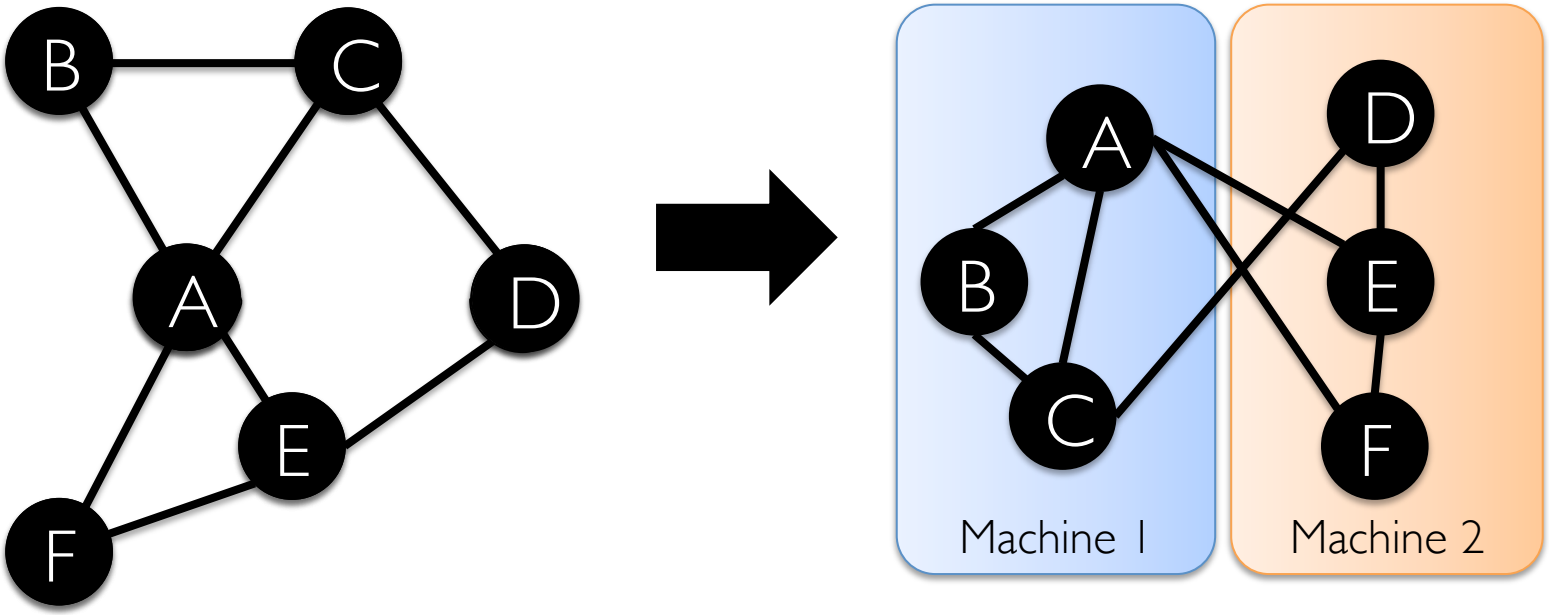
Domain specific graph partitioning

Spatial operations

Window operations

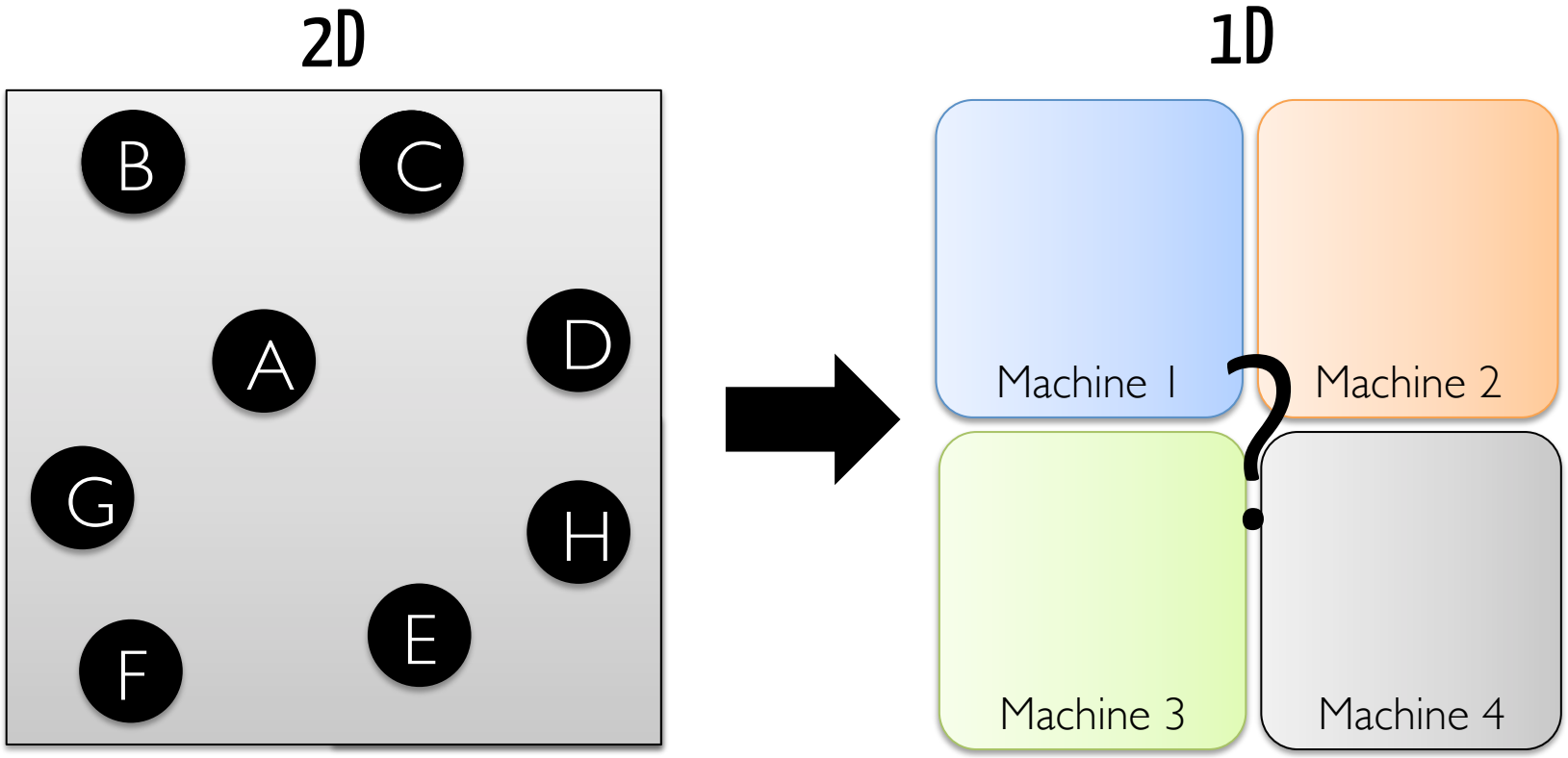
Graph Partitioning

Graph computation frameworks rely on partitioning to minimize communication & balance computation



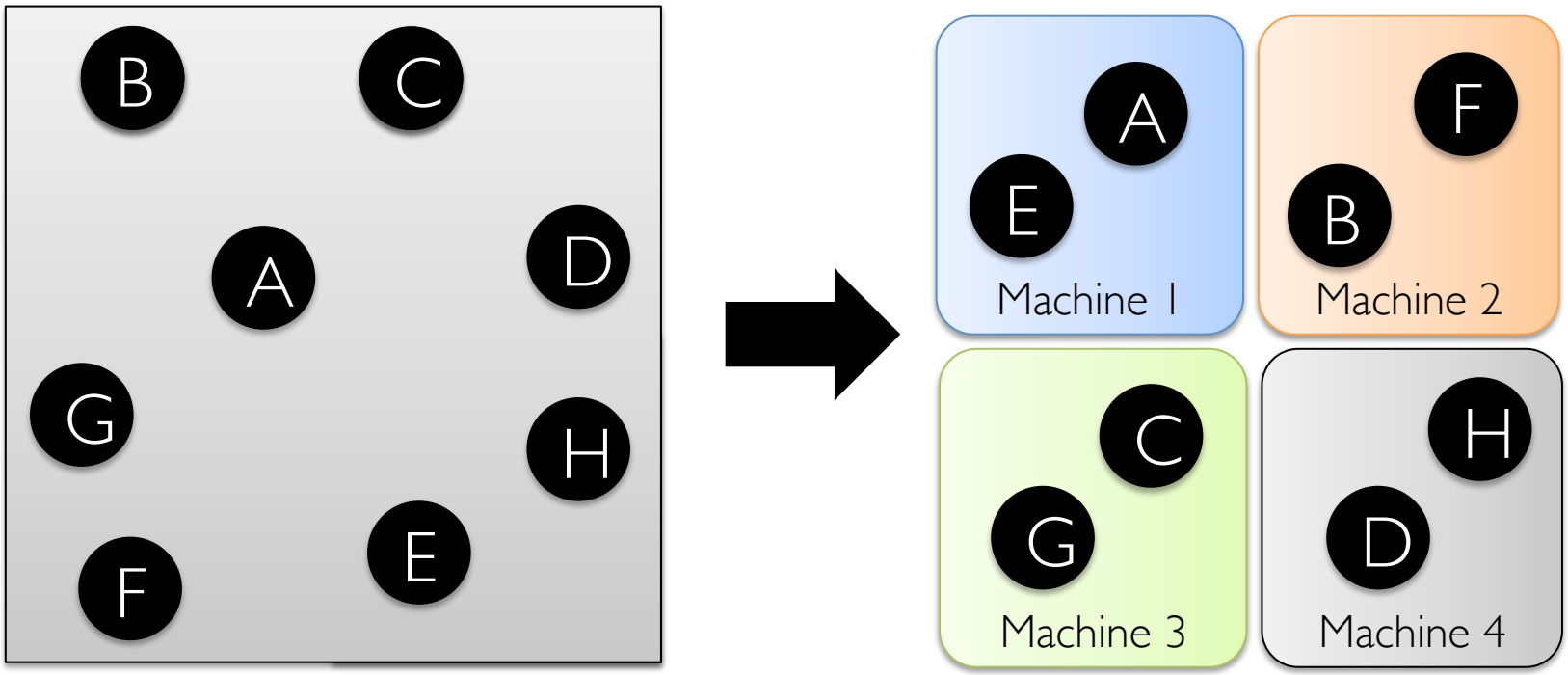
CellIQ Graph Partitioning

Partition geographically close-by entities



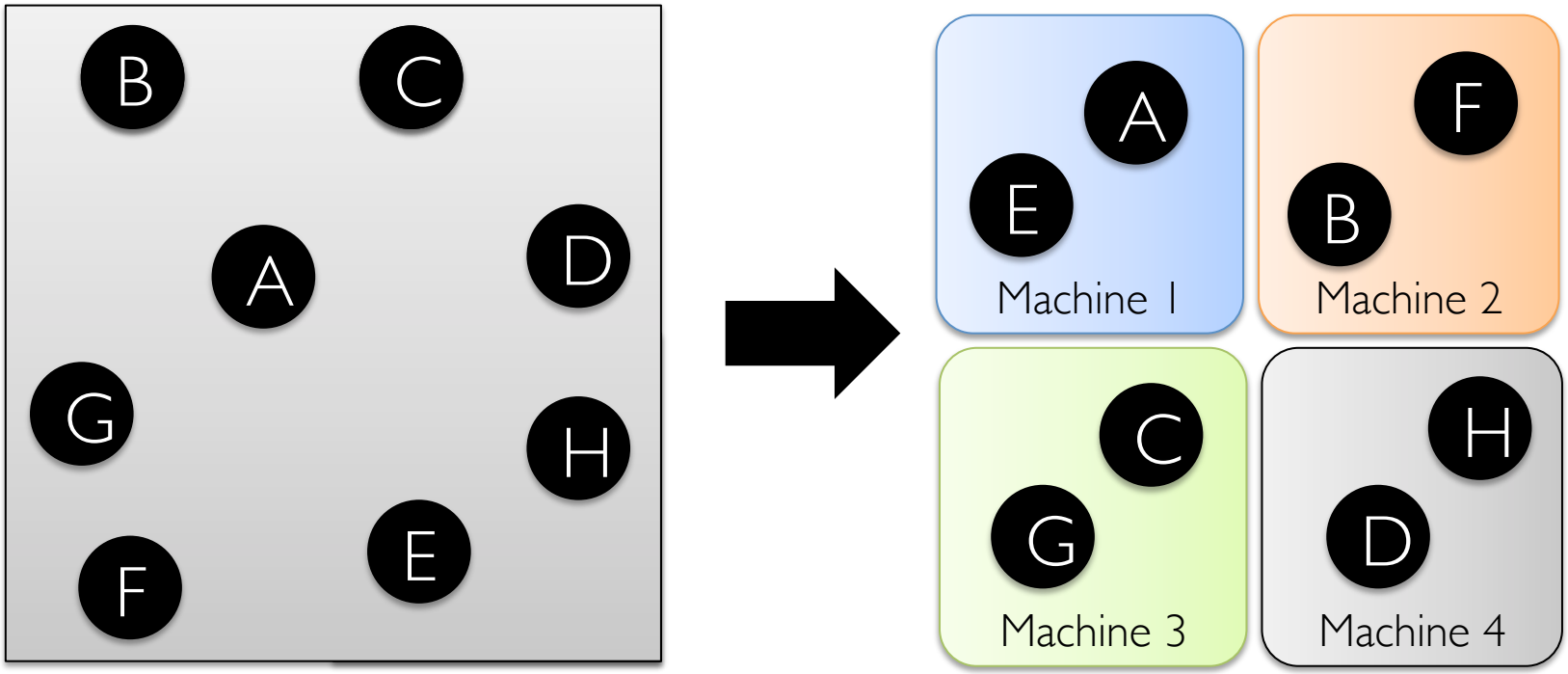
Graph Partitioning

Random (hashed) partitioning



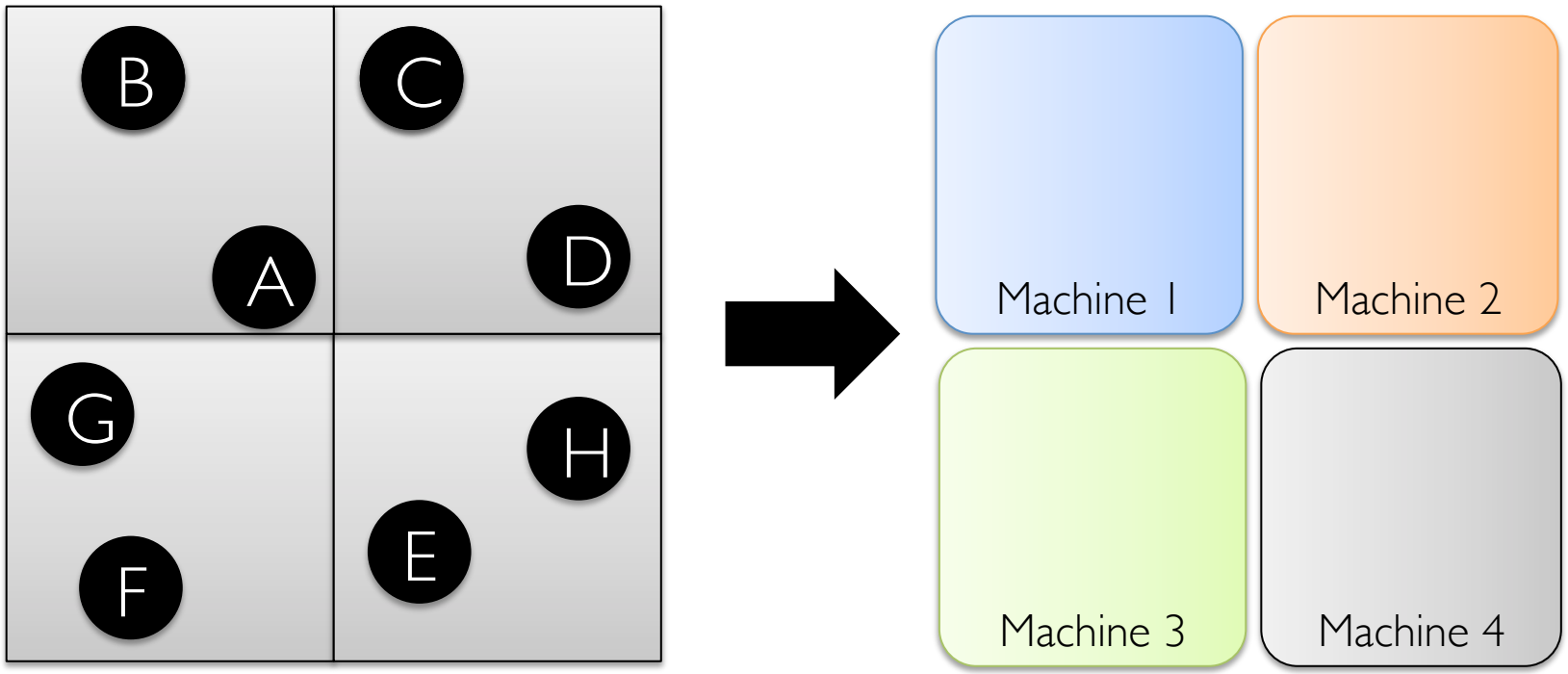
Graph Partitioning

Random (hashed) partitioning results in poor spatial locality



CellIQ Graph Partitioning

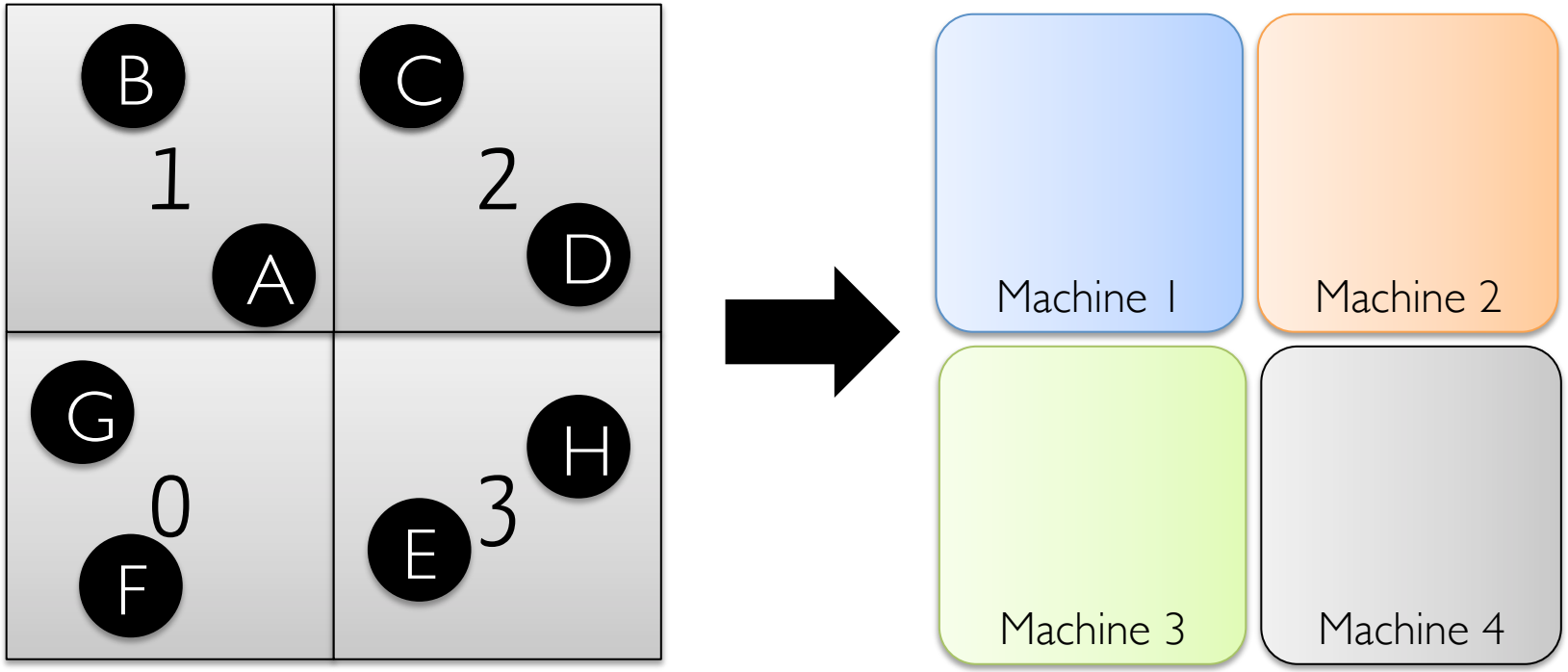
Uses Hilbert space-filling curves



CellIQ Graph Partitioning

Uses Hilbert space-filling curves

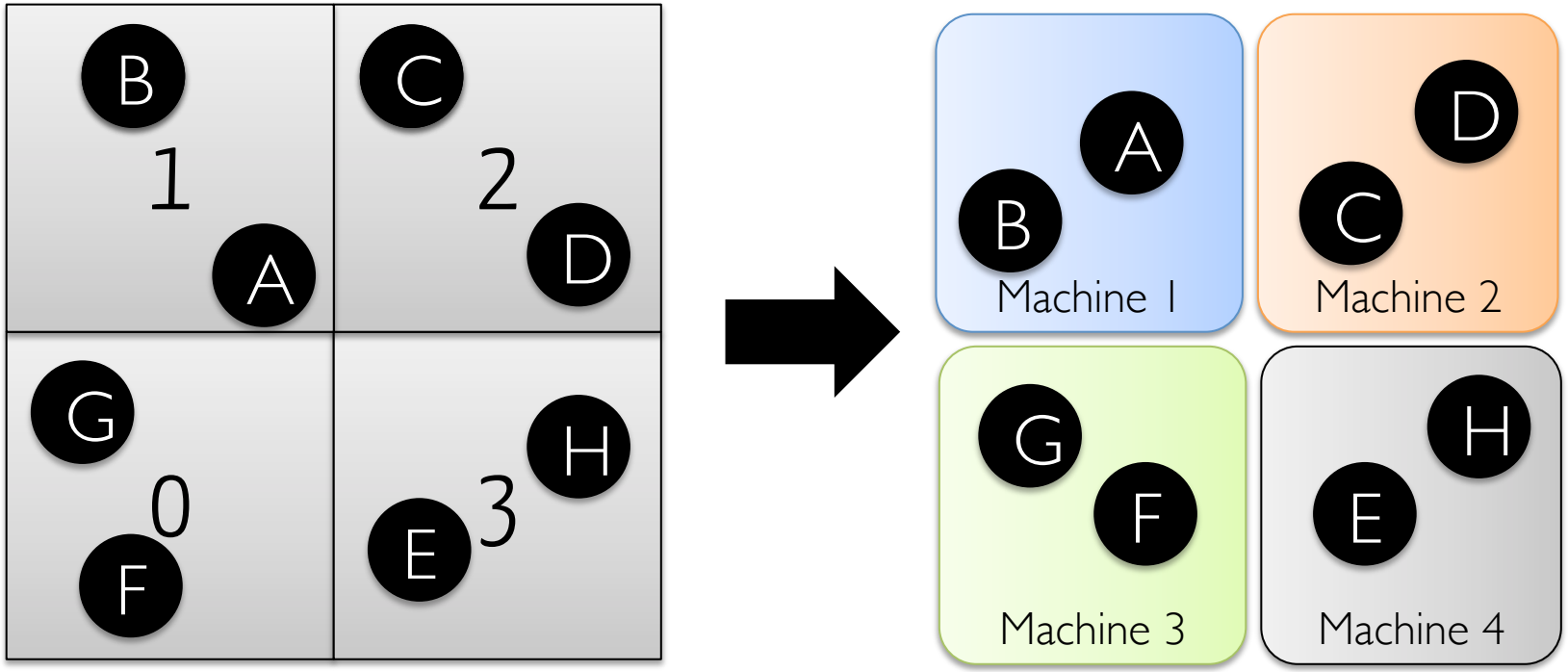
Use curve's distance as the 1-dimensional key



CellIQ Graph Partitioning

Uses Hilbert space-filling curves

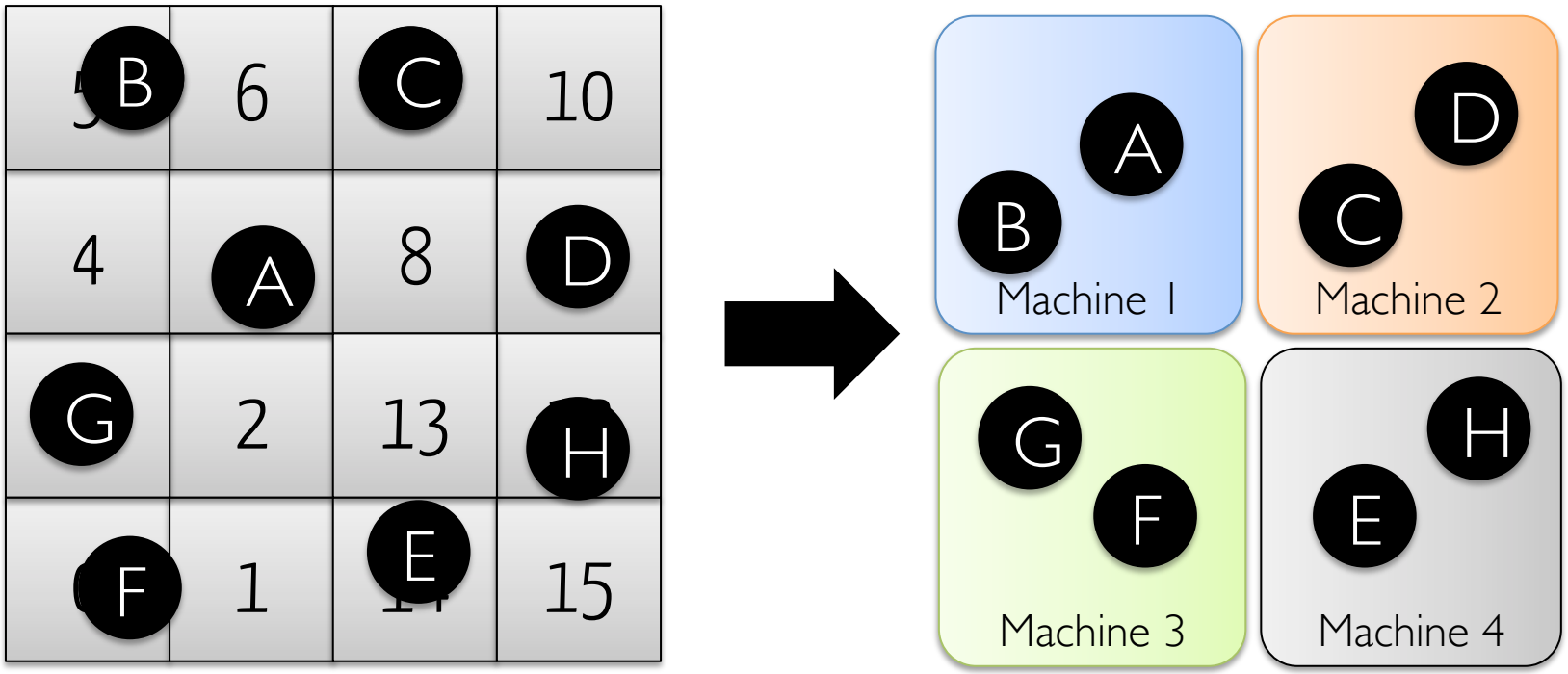
- Use curve's distance as the 1-dimensional key
- Range partition the key space



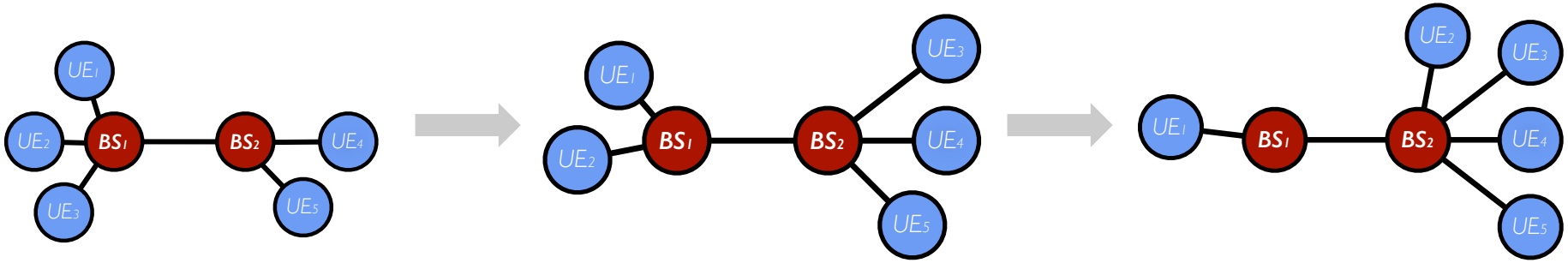
CellIQ Graph Partitioning

Uses Hilbert space-filling curves

- Use curve's distance as the 1-dimensional key
- Range partition the key space



Computational Model: GStreams



Domain specific graph partitioning

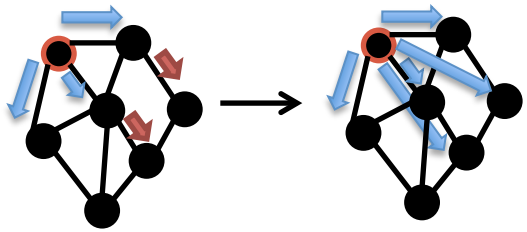
Spatial operations

Window operations

GeoGraph API

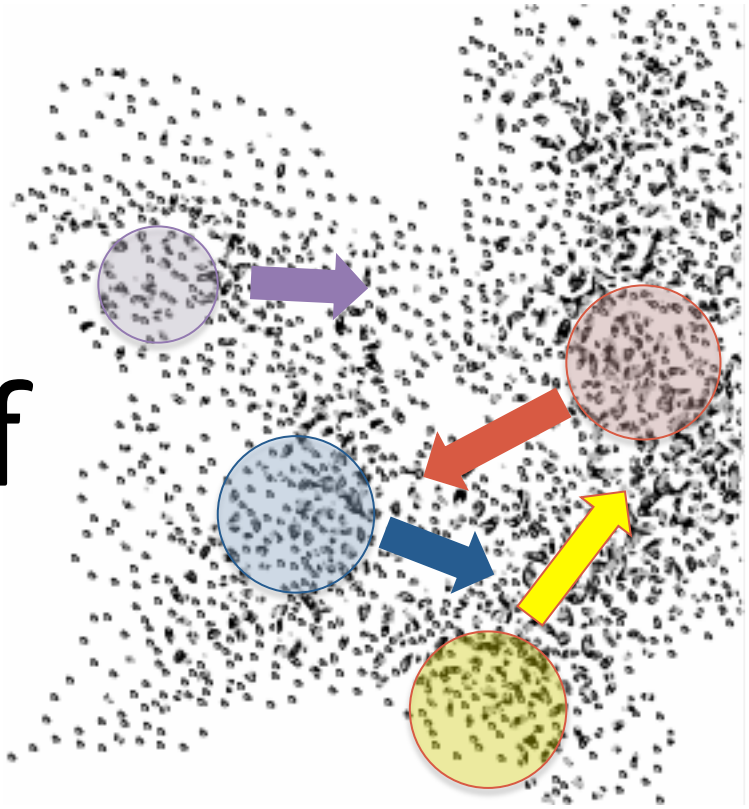
```
class GeoGraph[V, E] {
  // Broadcast a message to all
  // vertices within a radius
  def sendMsg(radius)

  // Create a spatially aggregated
  // graph by combining vertices
  // and edges
  def spatialAG(reduceV: (V, V) => V,
                reduceE: (E, E) => E)
}
```

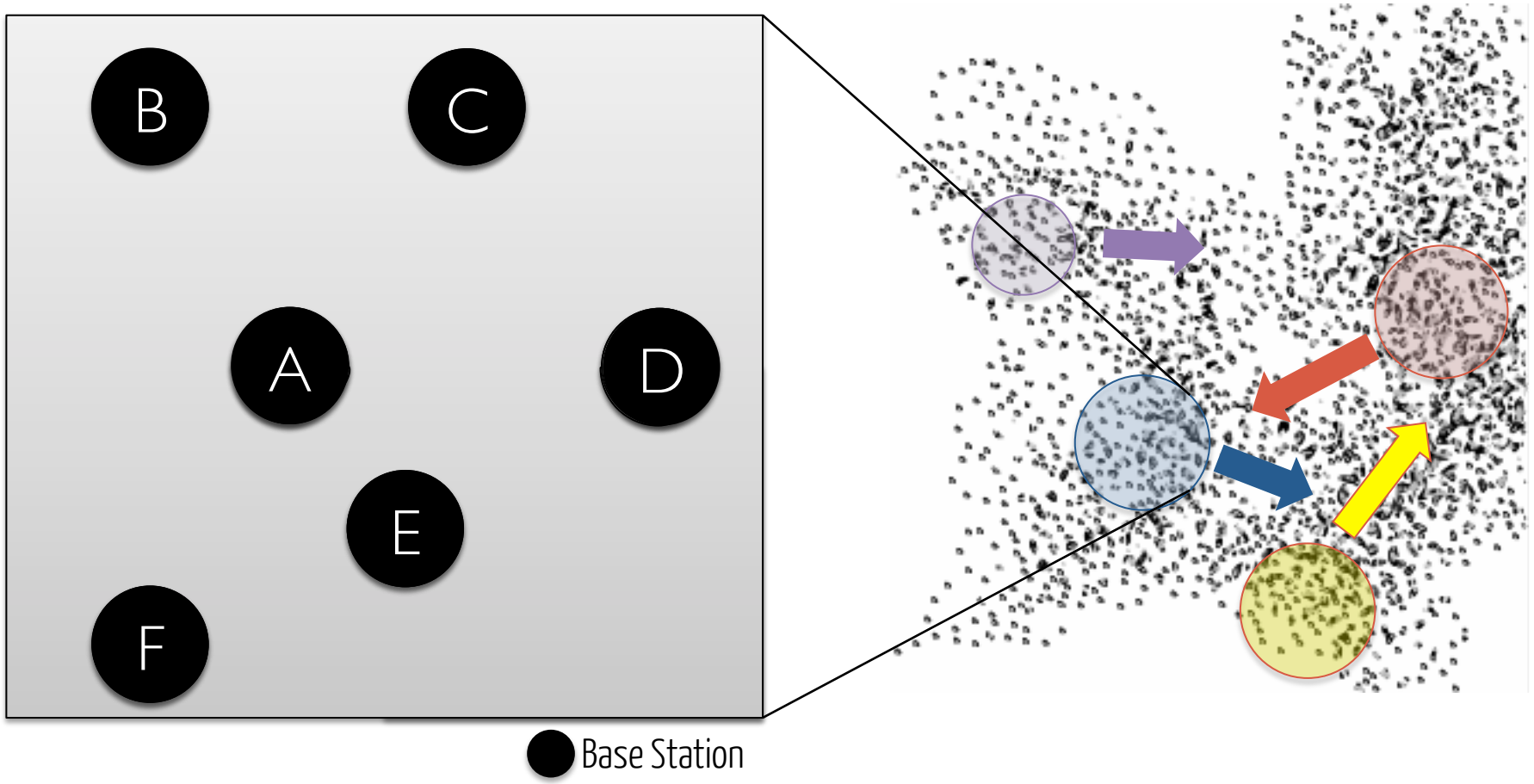


Tracking user traffic gradients

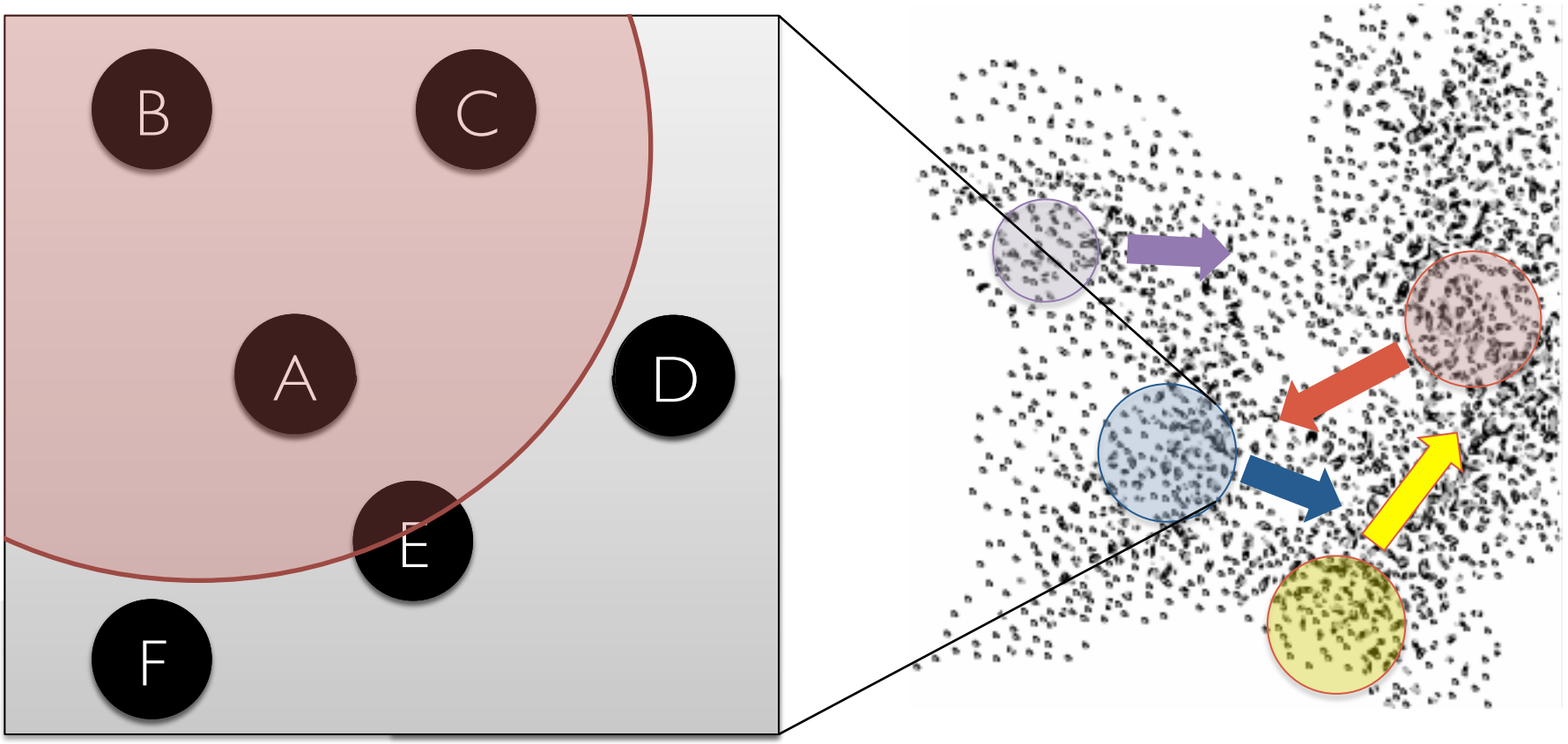
Goal: Detect and track direction of movement of user groups



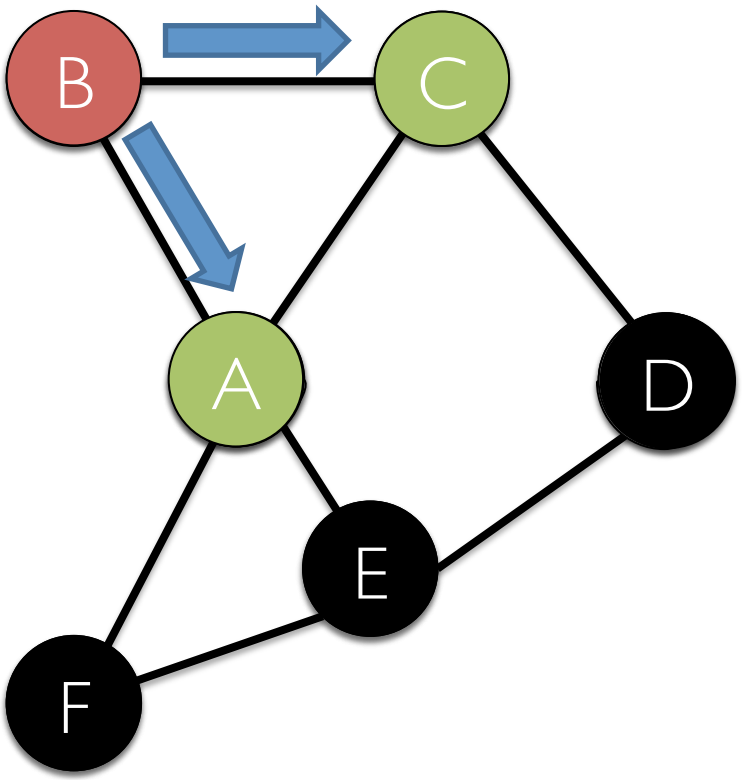
Tracking user traffic gradients



Tracking user traffic gradients

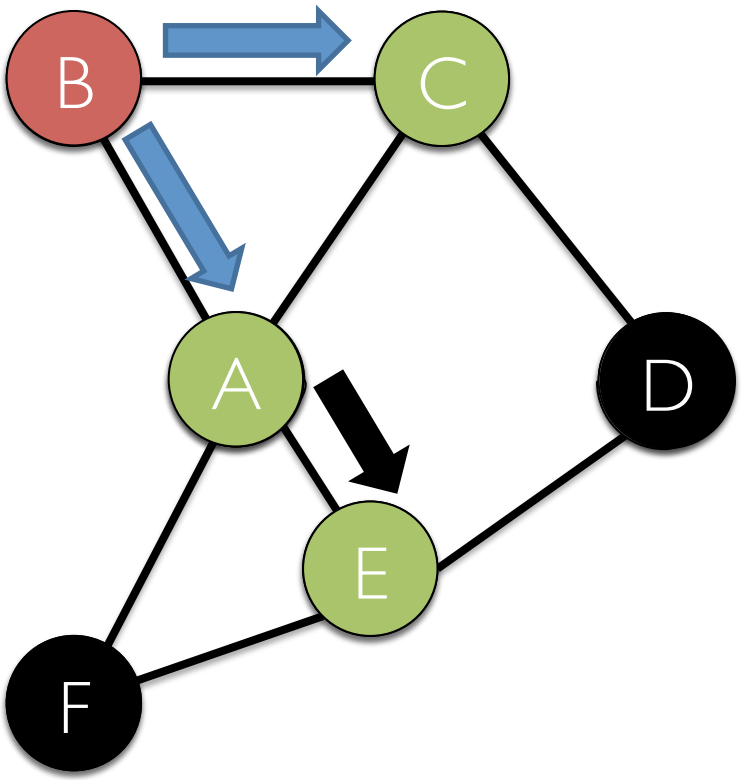


Tracking user traffic gradients



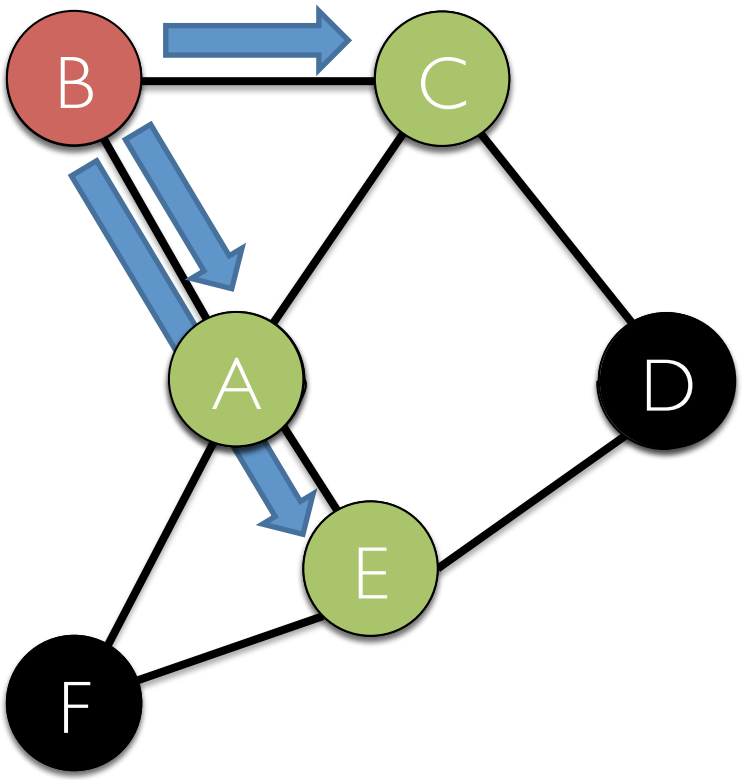
Hop-by-hop propagation

Tracking user traffic gradients



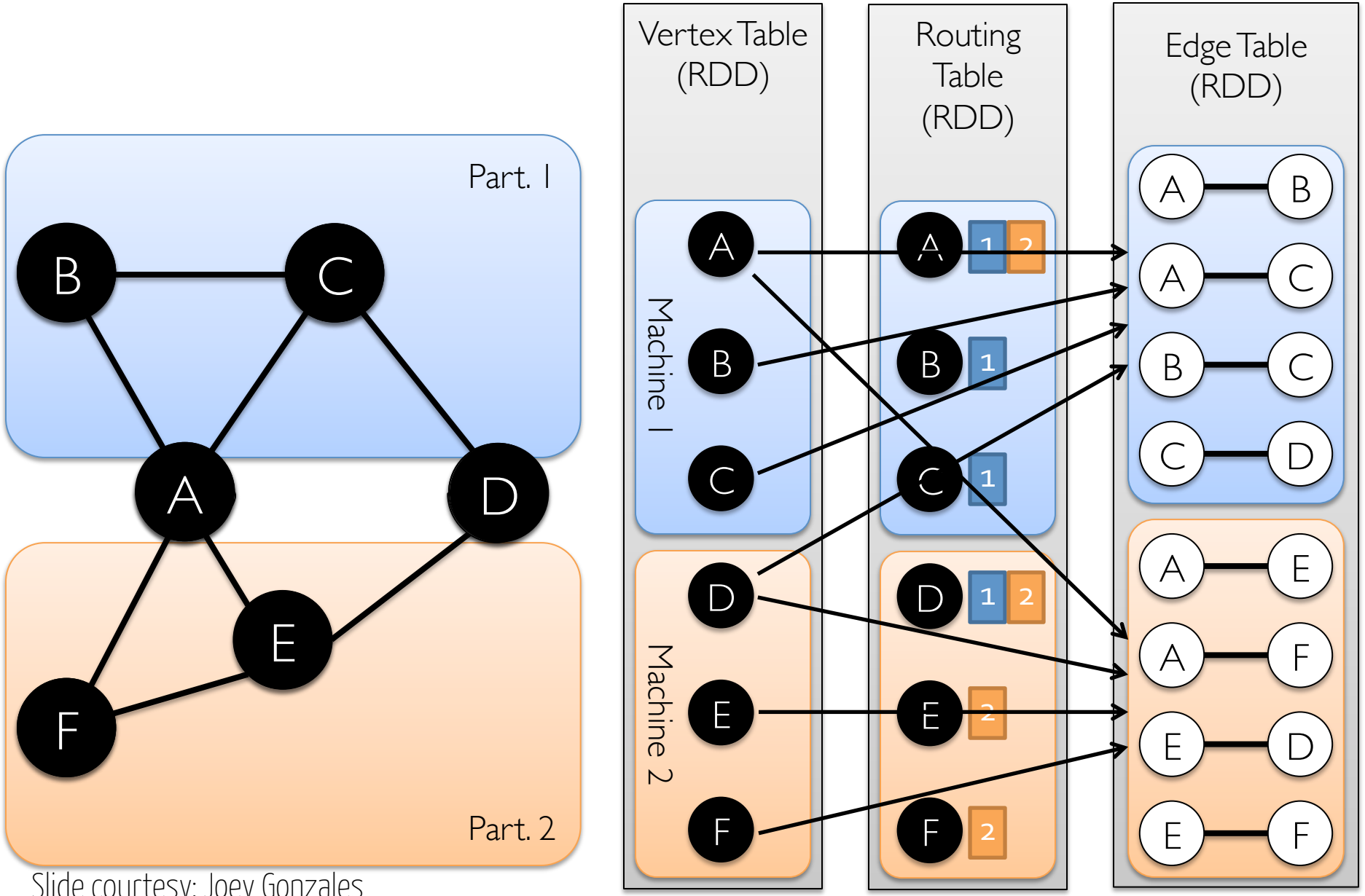
Hop-by-hop propagation is inefficient

Tracking user traffic gradients



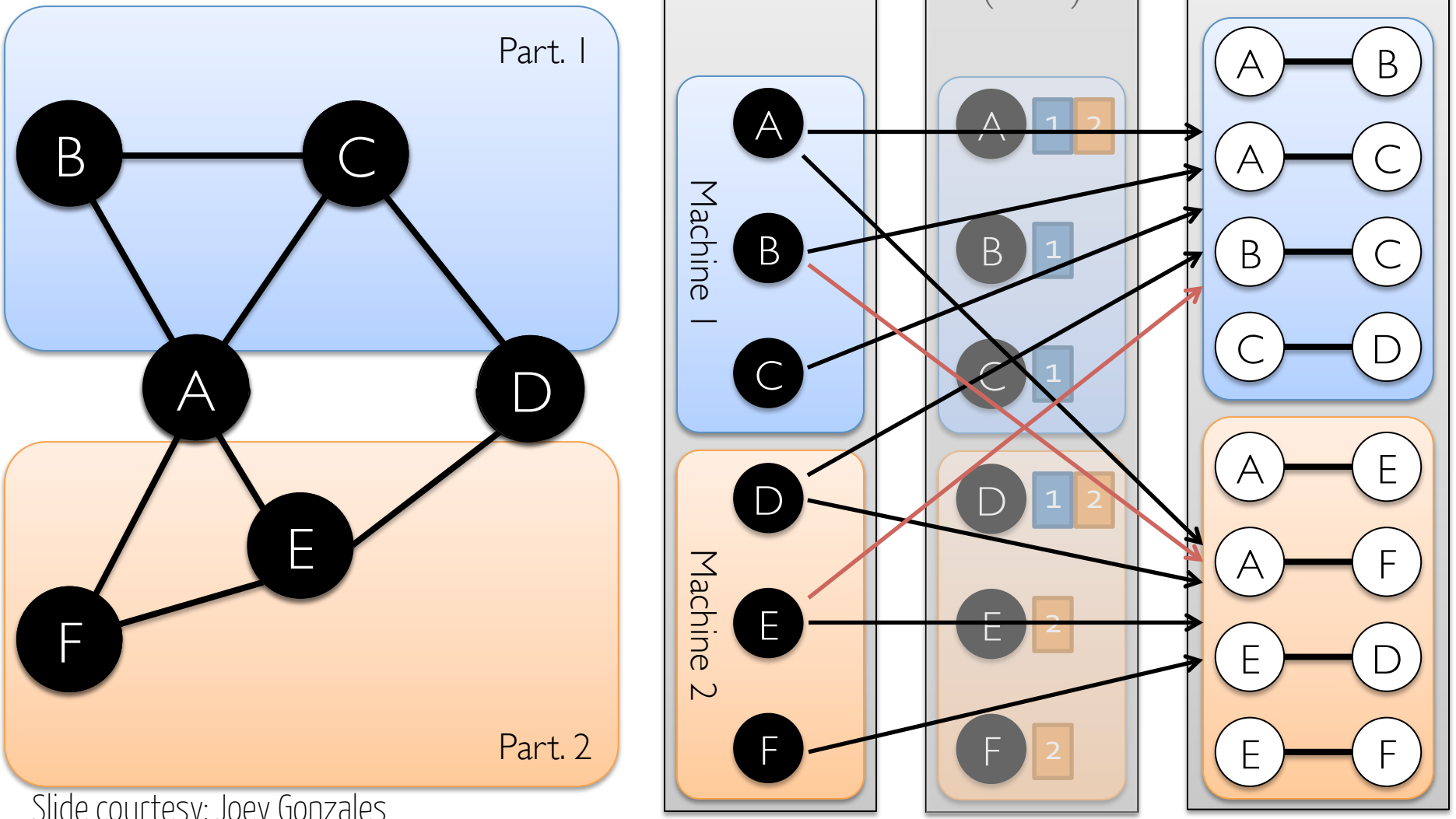
Instead, CellIQ enables radius based broadcast

Routing Table in GraphX enables Multicast



Slide courtesy: Joey Gonzales

Can compute destination partitions easily due to the use of geo-partitioner

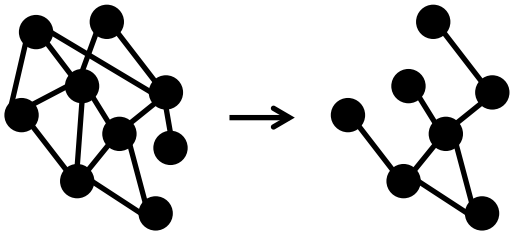


Slide courtesy: Joey Gonzales

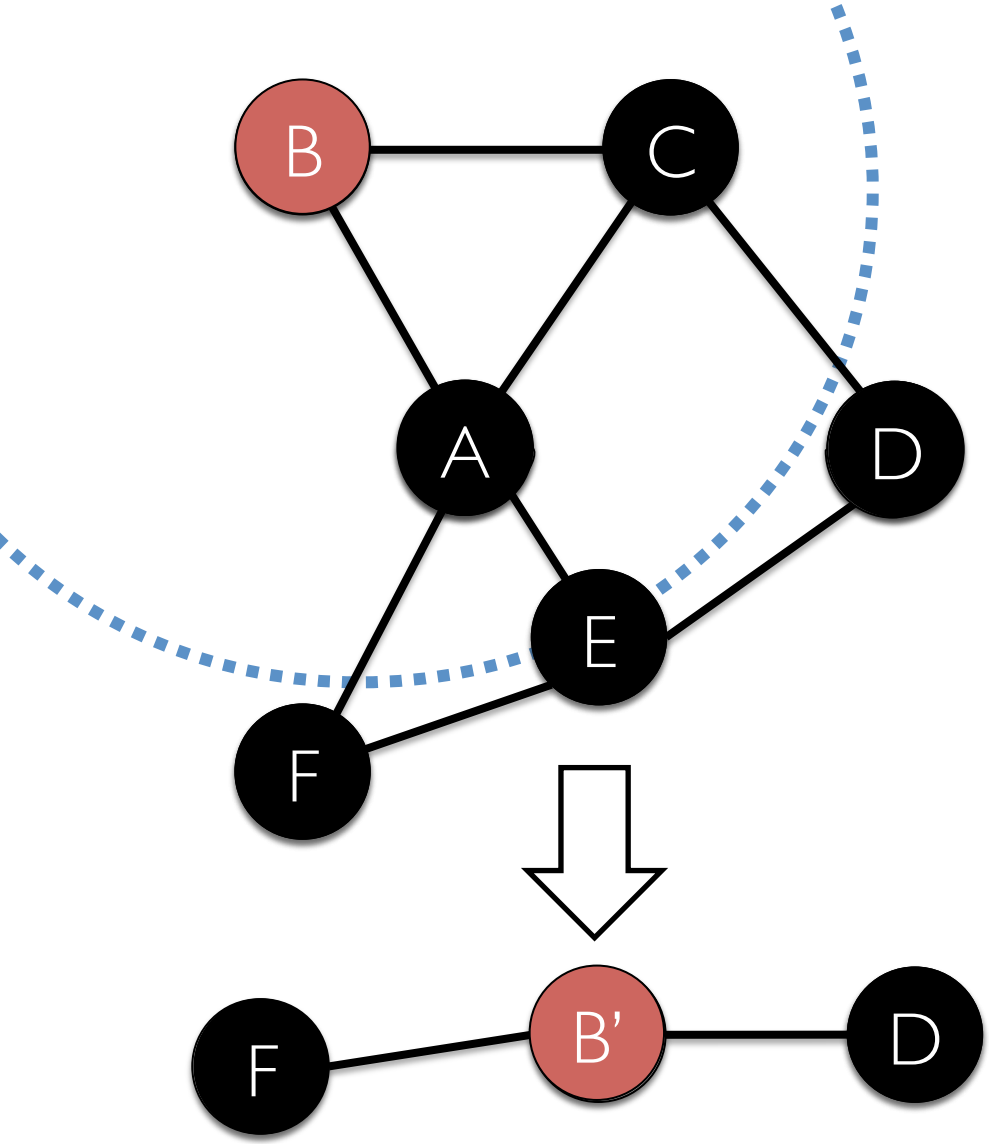
GeoGraph API

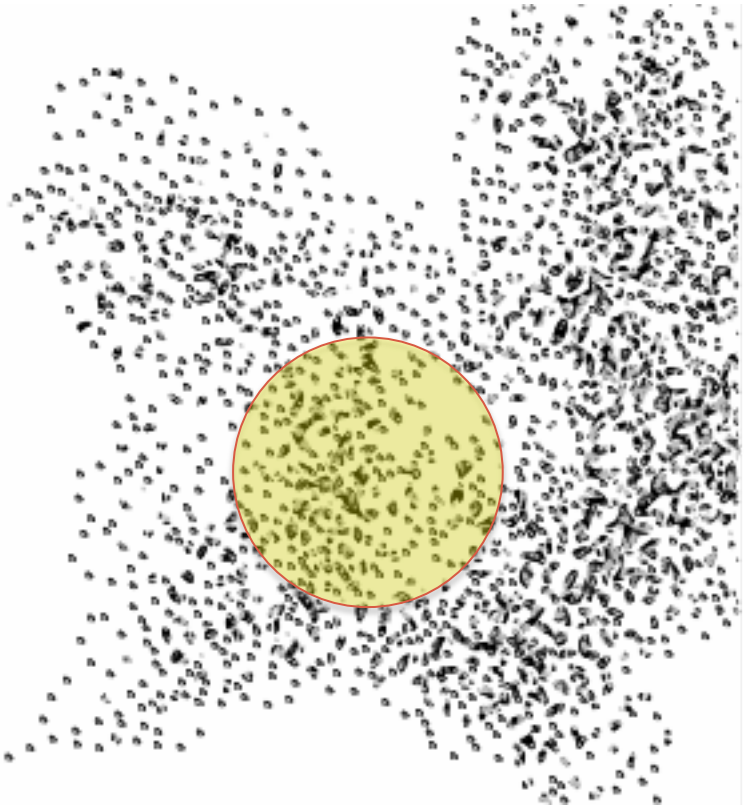
```
class GeoGraph[V, E] {
  // Broadcast a message to all
  // vertices within a radius
  def sendMsg(radius)


  // Create a spatially aggregated
  // graph by combining vertices
  // and edges
  def spatialAG(reduceV: (V, V) => V,
                reduceE: (E, E) => E)
}
```

Spatial Clustering



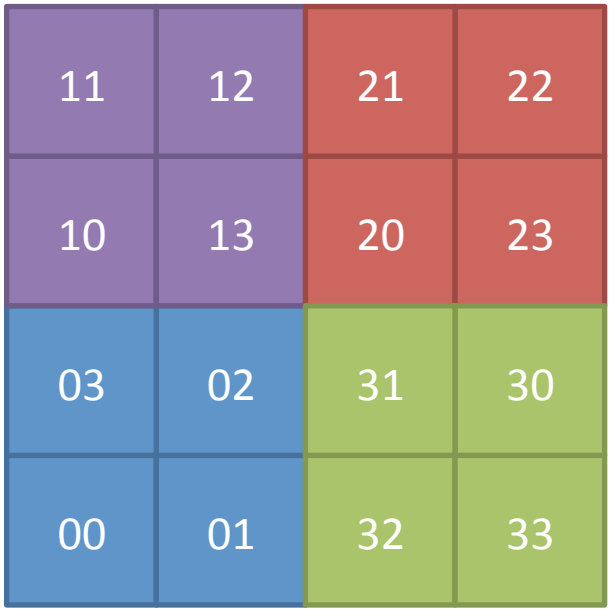


Goal: Combine spatially close-by vertices

Spatial Clustering

Two ways to enable spatial aggregation:

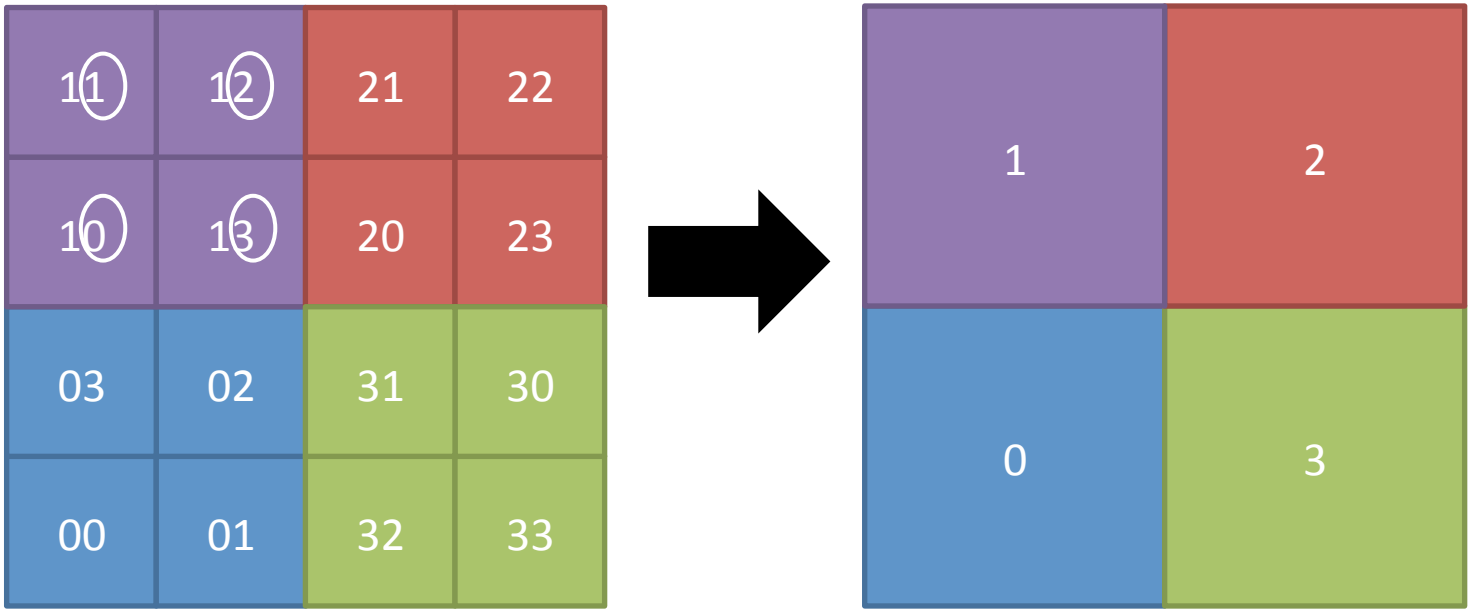
- Using a (supplied) field in properties
- Leverage geo partitioner



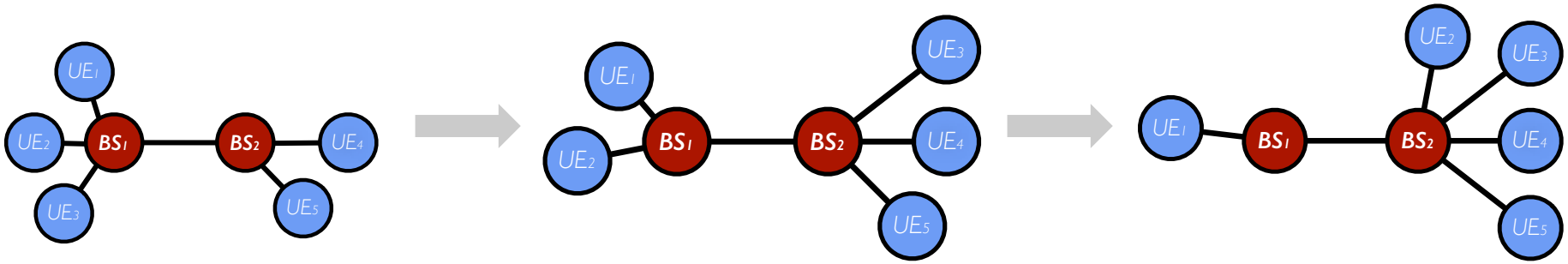
Spatial Clustering

Two ways to enable spatial aggregation:

- Using a (supplied) field in properties
- Leverage geo partitioner



Computational Model: GStreams



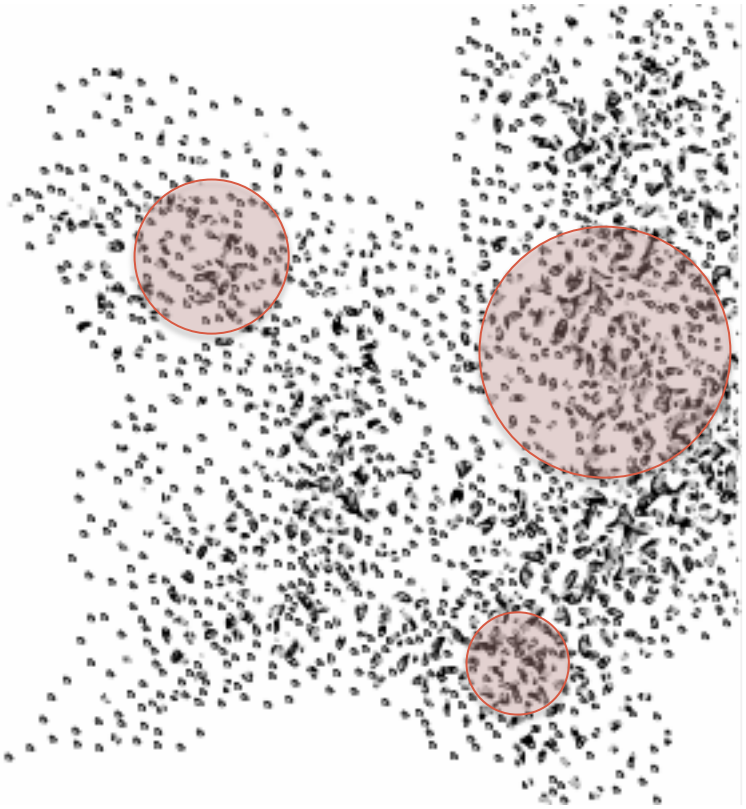
Domain specific graph partitioning

Spatial operations

Window operations

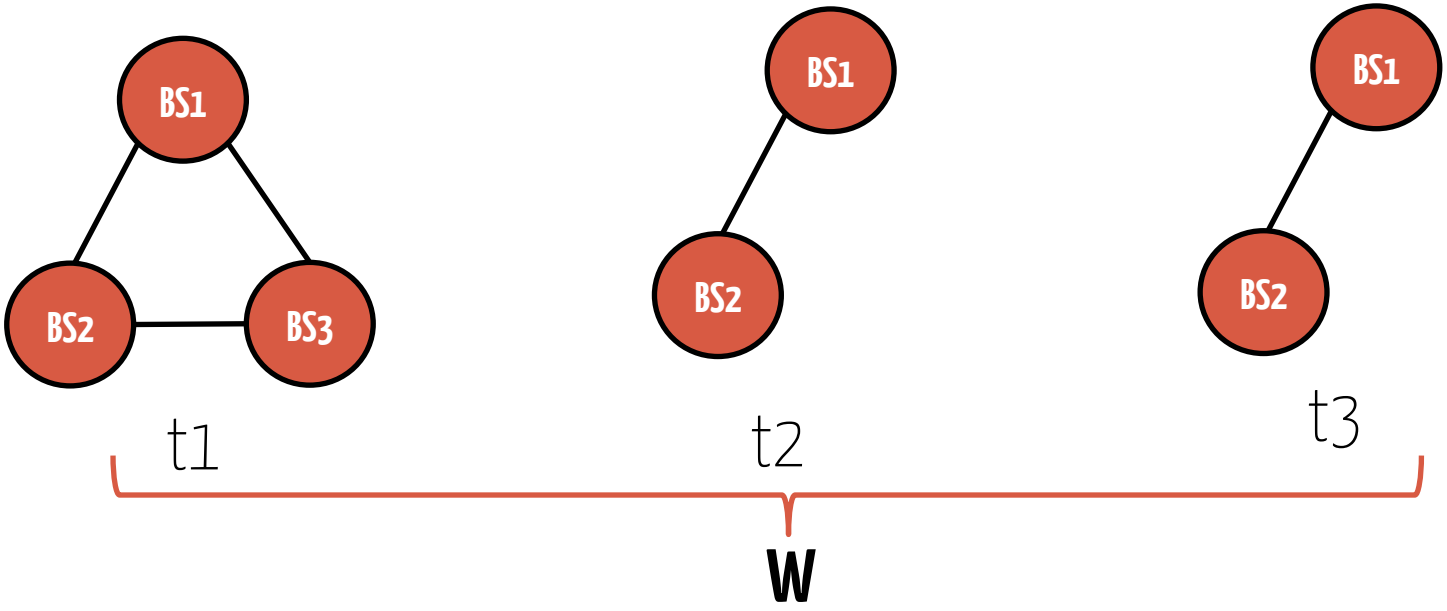
Tracking Persistent Hotspots

Goal: Detect and track groups of base stations with high traffic volume



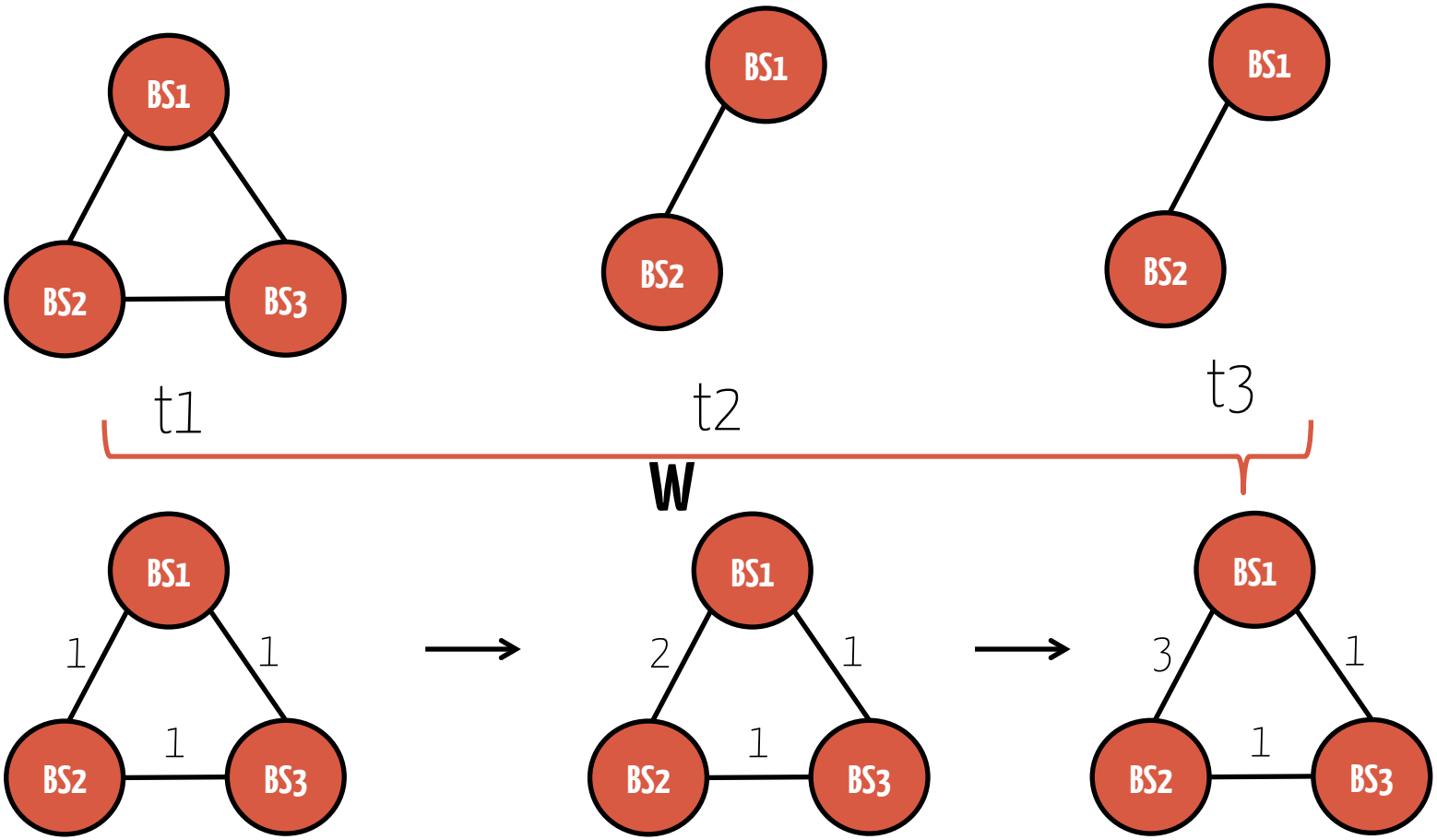
Equivalent to finding connected components

Tracking Persistent Hotspots



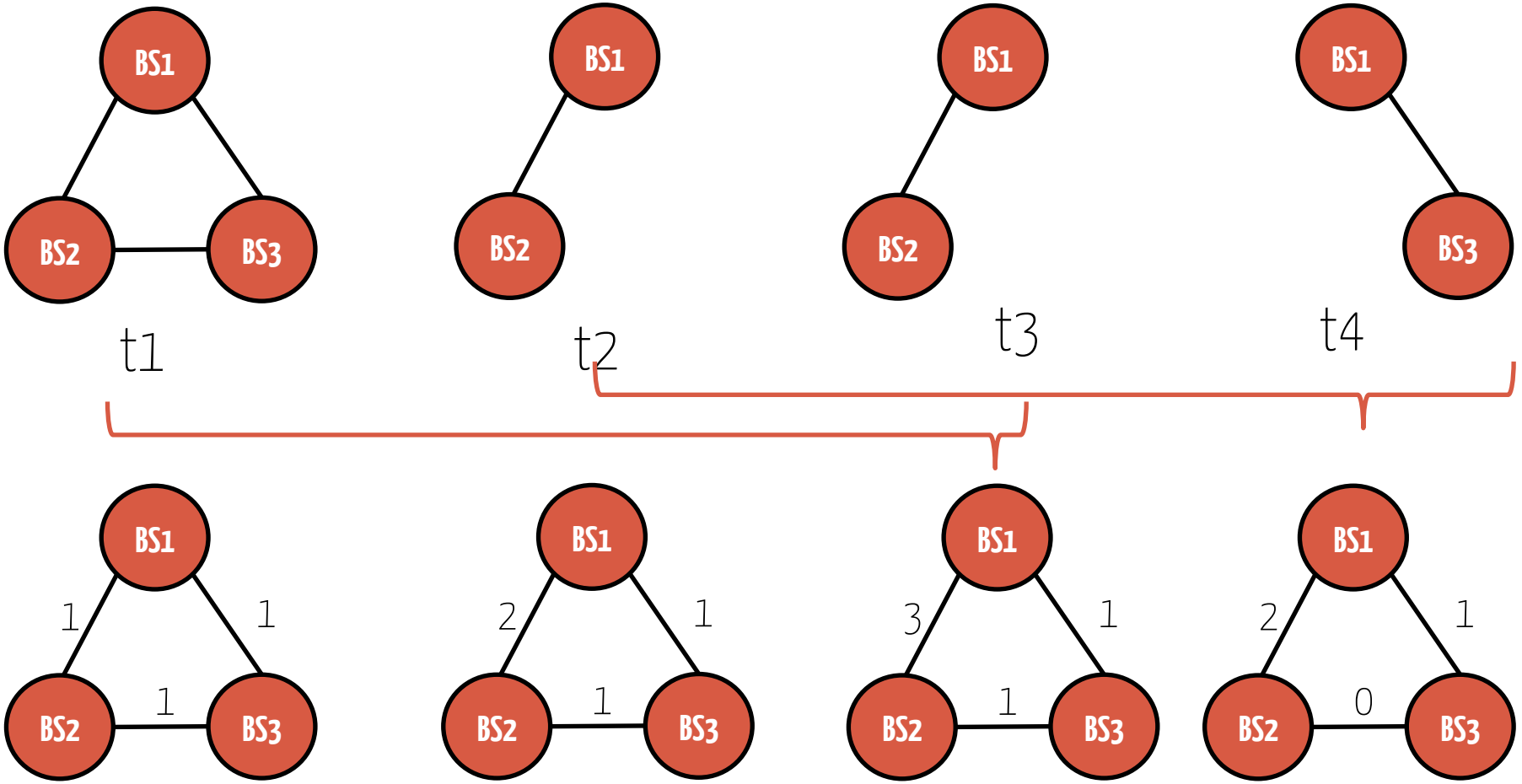
Combining graphs at the end of the window results in many join operations (inefficient)

Tracking Persistent Hotspots



Apply incremental updates to a cumulative graph

Tracking Persistent Hotspots



Apply differential updates to a cumulative graph

GStream API

```
class GStream[V, E] {

  def graphReduceByWindow(
    reduceFunc(Graph[V, E], Graph[V, E],
               fv: (V, V) => V,
               fe: (E, E) => E): Graph[V, E],
    invReduceFunc(Graph[V, E], Graph[V, E],
               fv: (V, V) => V,
               fe: (E, E) => E): Graph[V, E],
    windowDuration, slideDuration)
}
```

`graphReduceByWindow`

- Implemented using Spark's `cogroupedRDD`
- Two default reduce functions: graph `intersection` and `union`
- Further optimizations:
  - Co-partition graphs from multiple batches
  - Reuse indices and routing tables for graphs in the same window

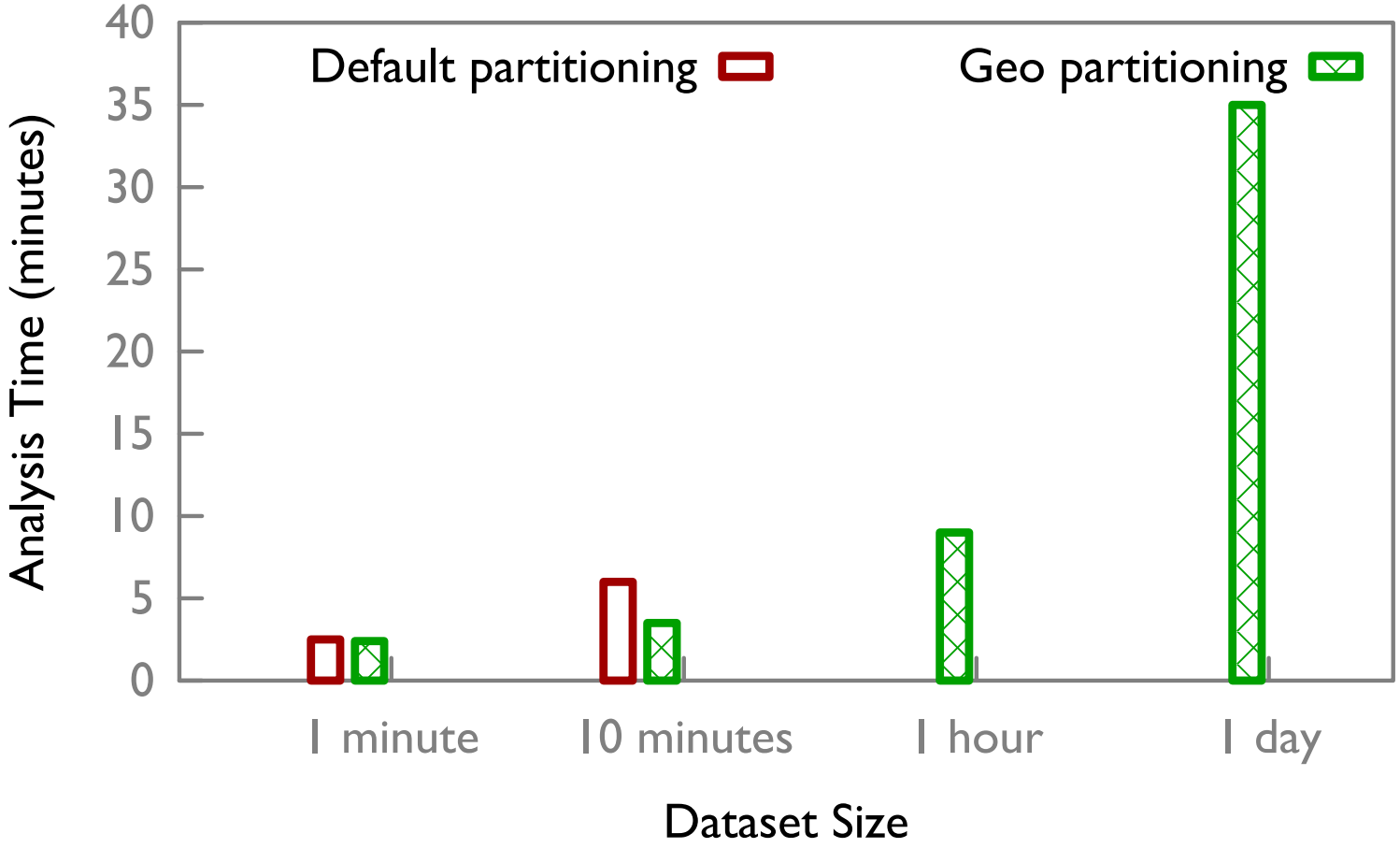
More details in the paper!

How does CellIQ perform?

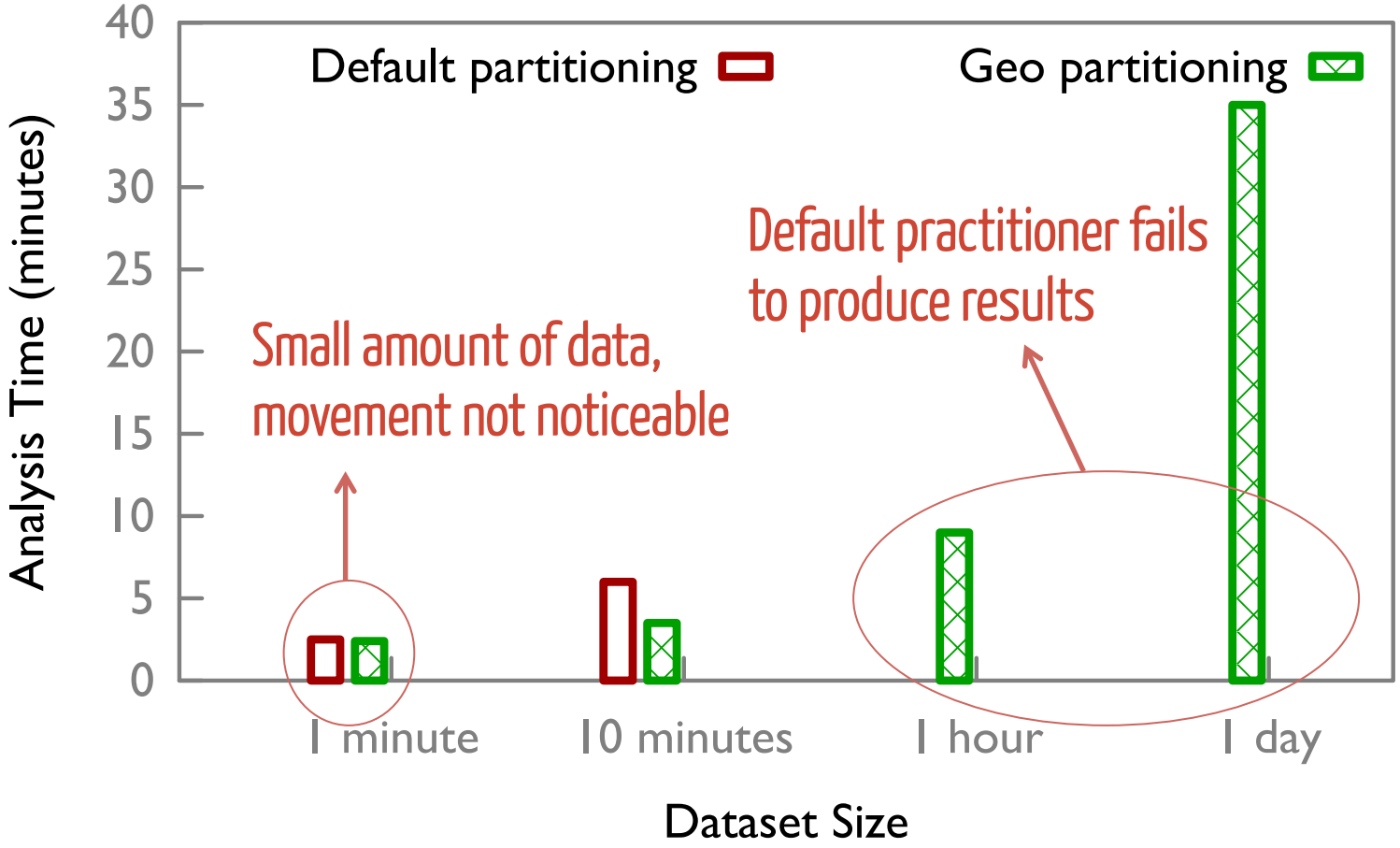
Evaluation Setup

- LTE control plane data from a major cellular network operator
- 1 million+ subscribers, live network
- 2 TB data from 1 week
  - 1 file per minute, 750k records, 100s of fields/line
  - 10 collection points, 10 hours per day
- Implemented several analysis tasks

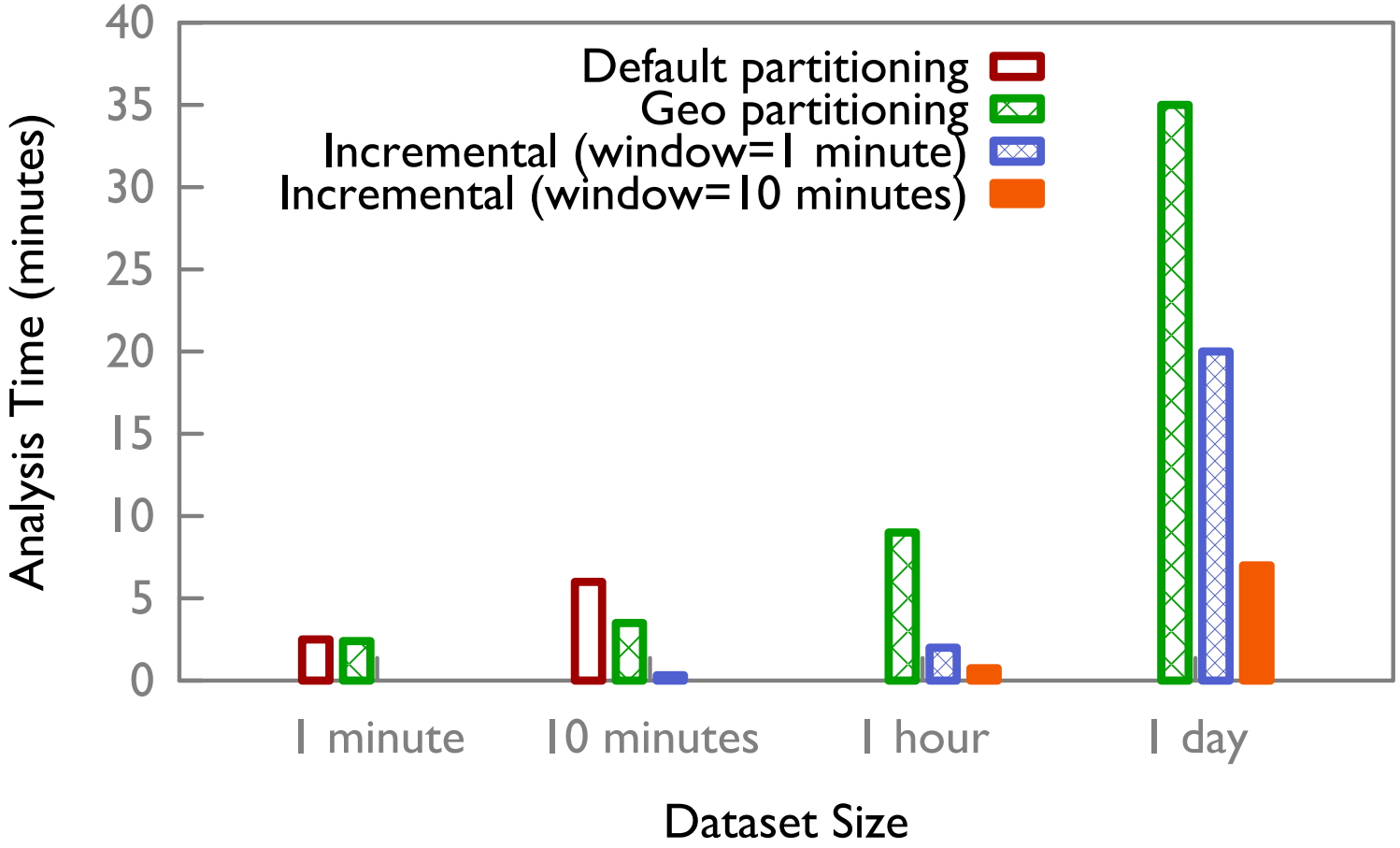
Benefits of Geo-partitioning



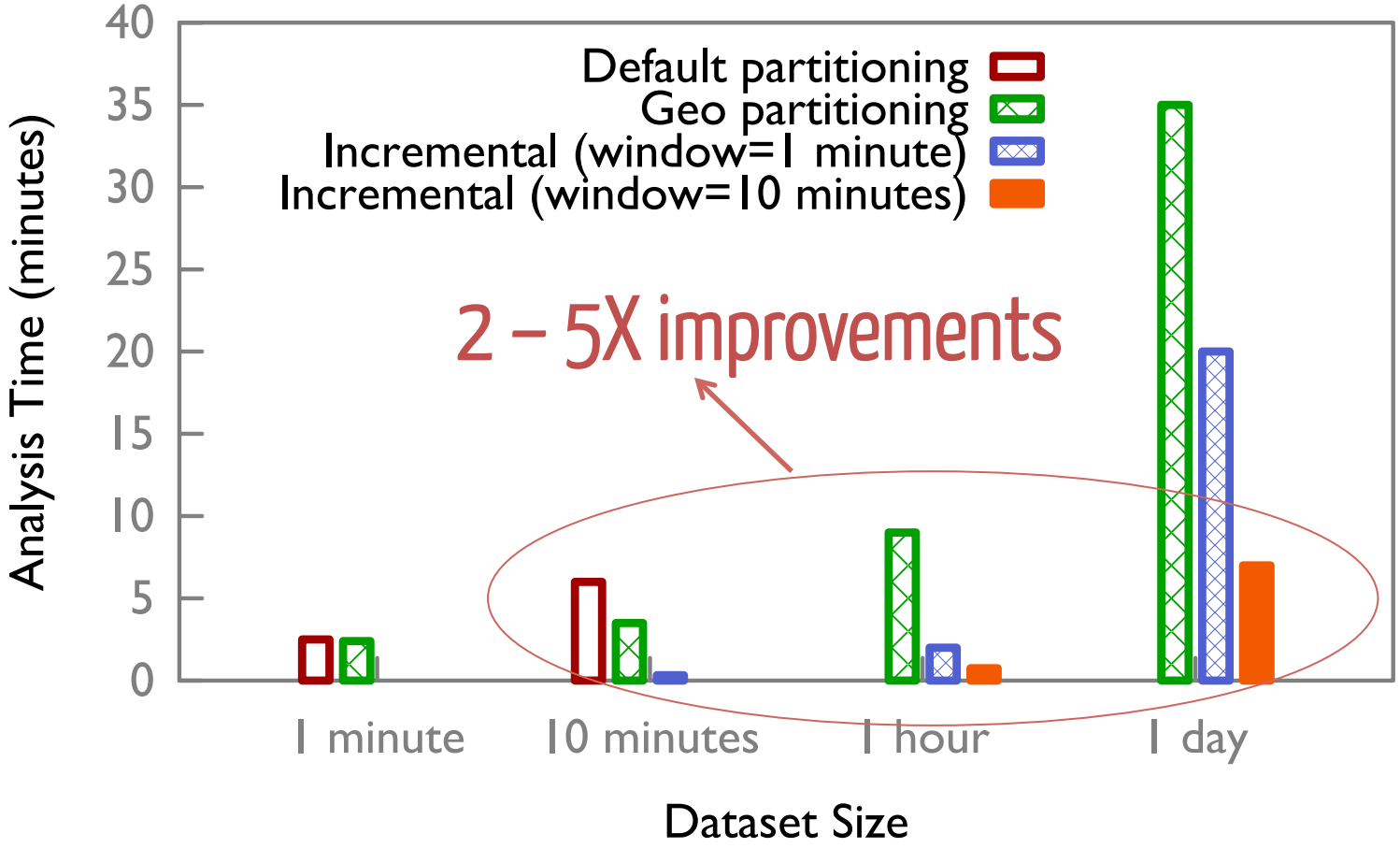
Benefits of Geo-partitioning



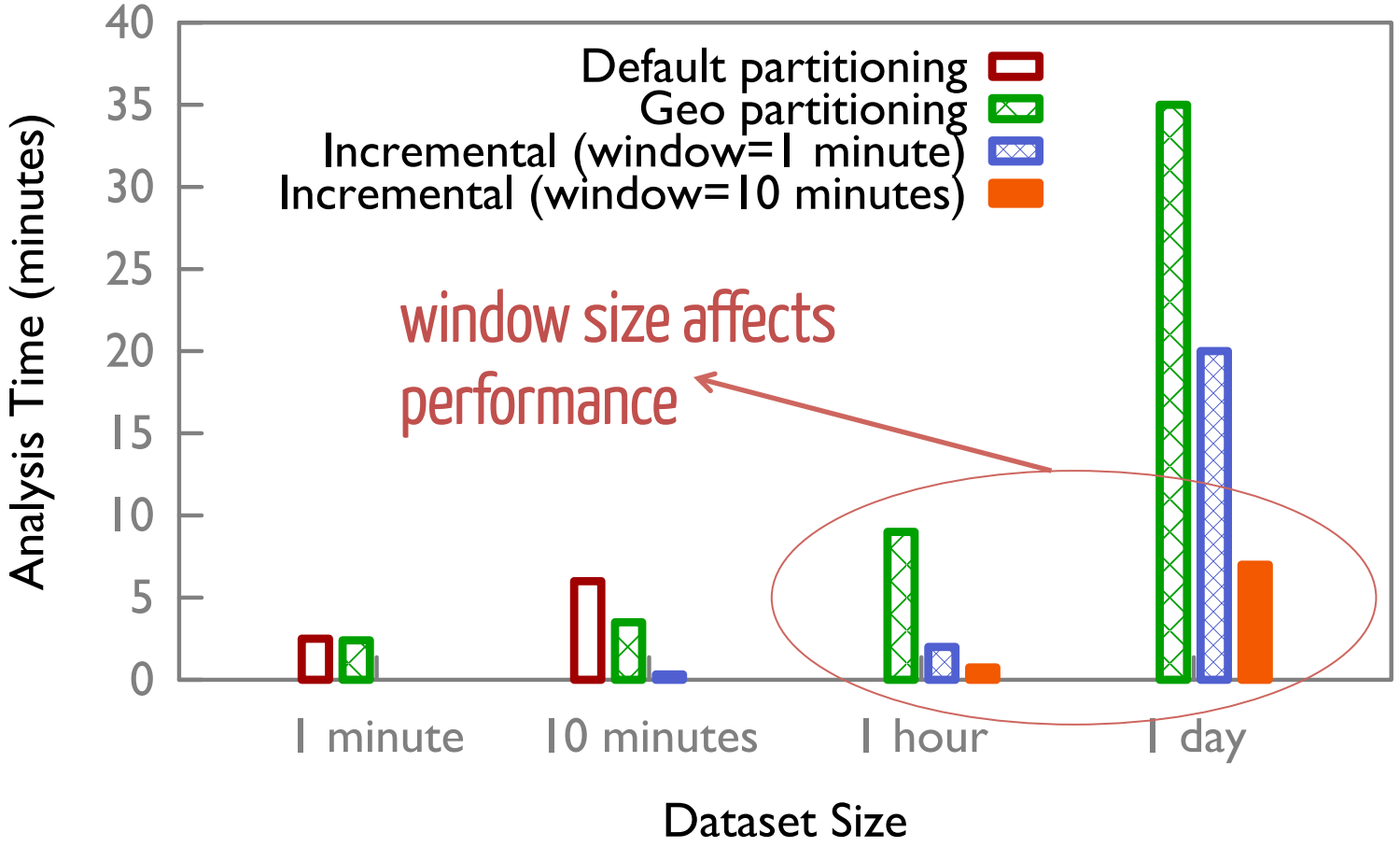
Benefits of Incremental Updates



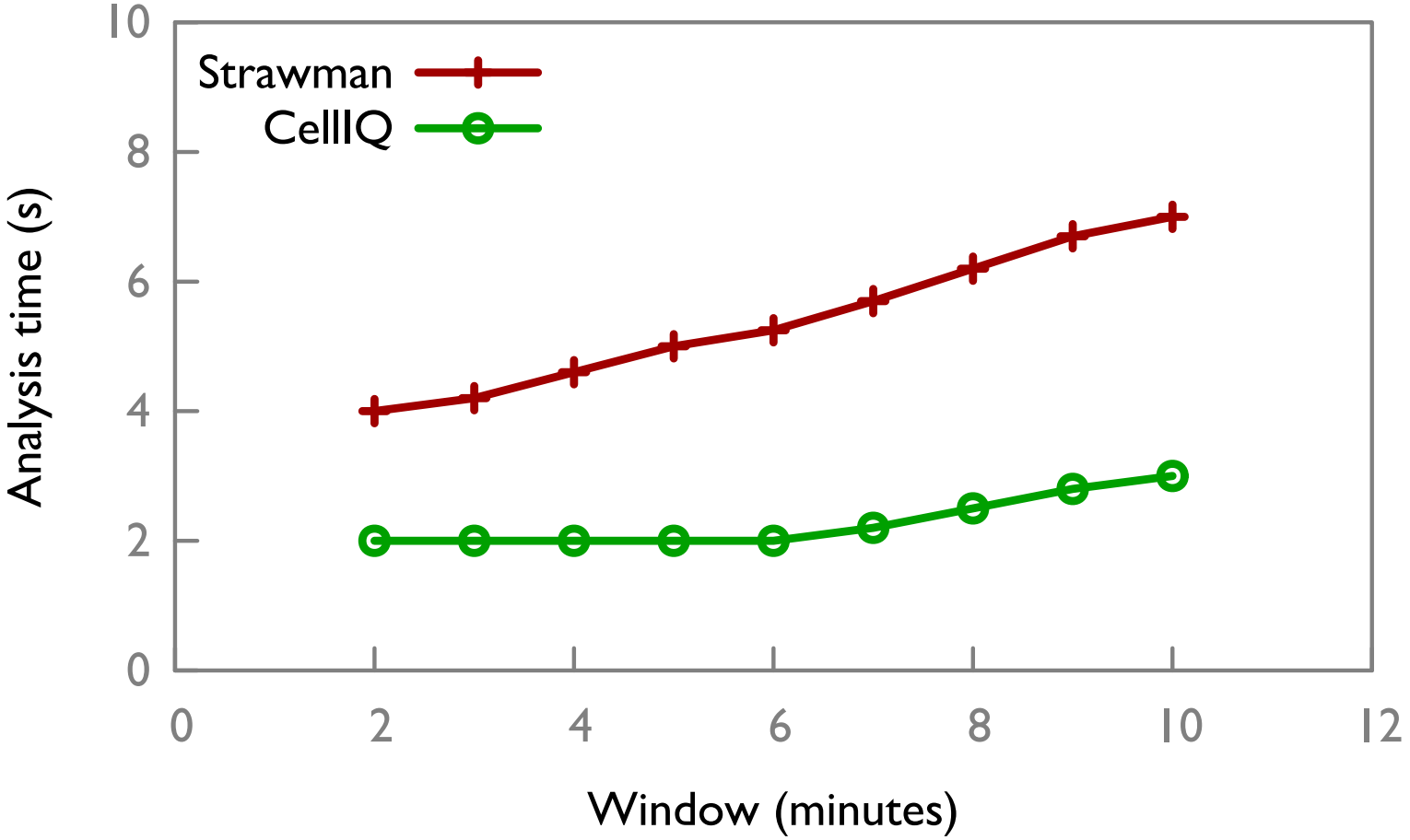
Benefits of Incremental Updates



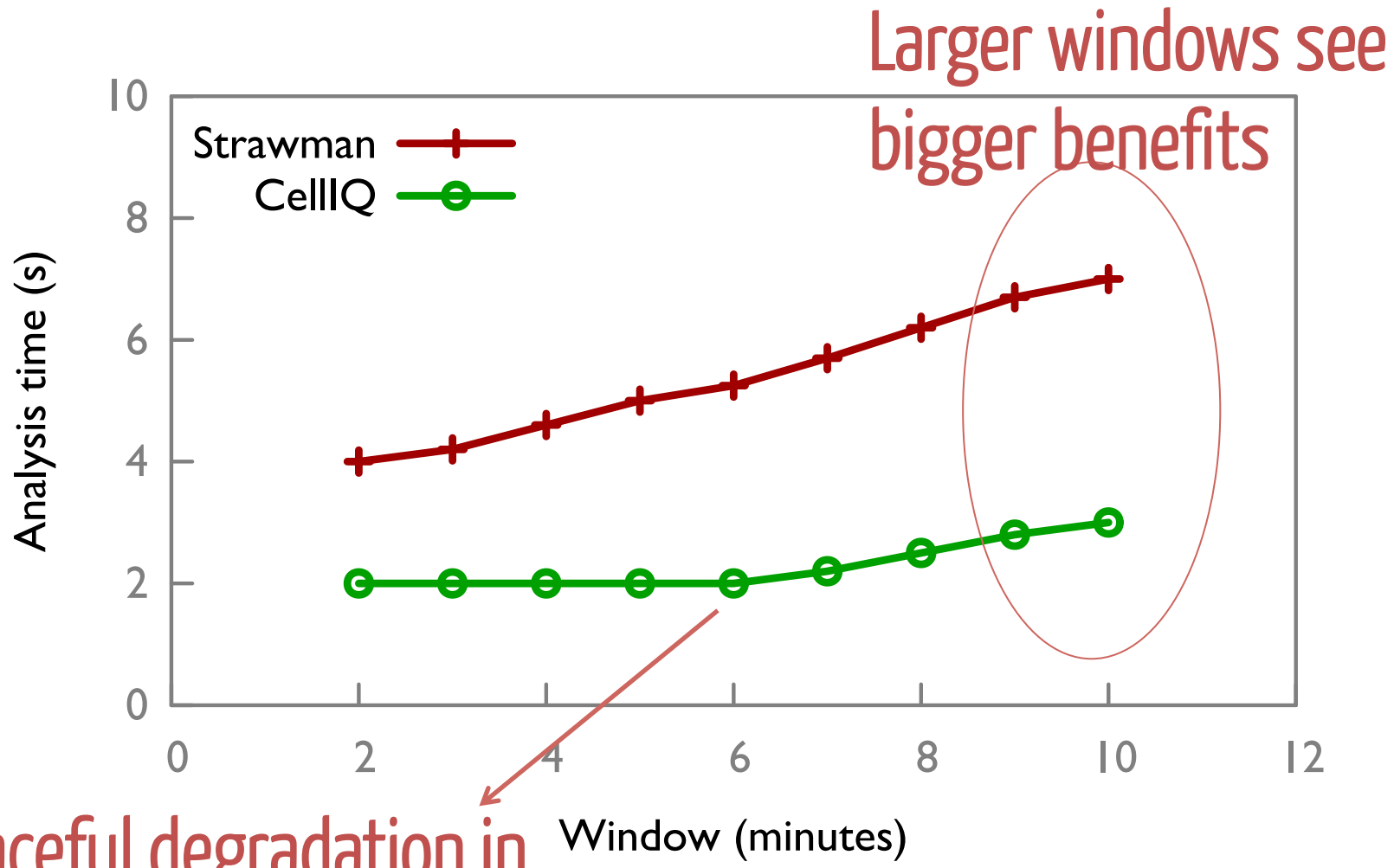
Benefits of Incremental Updates



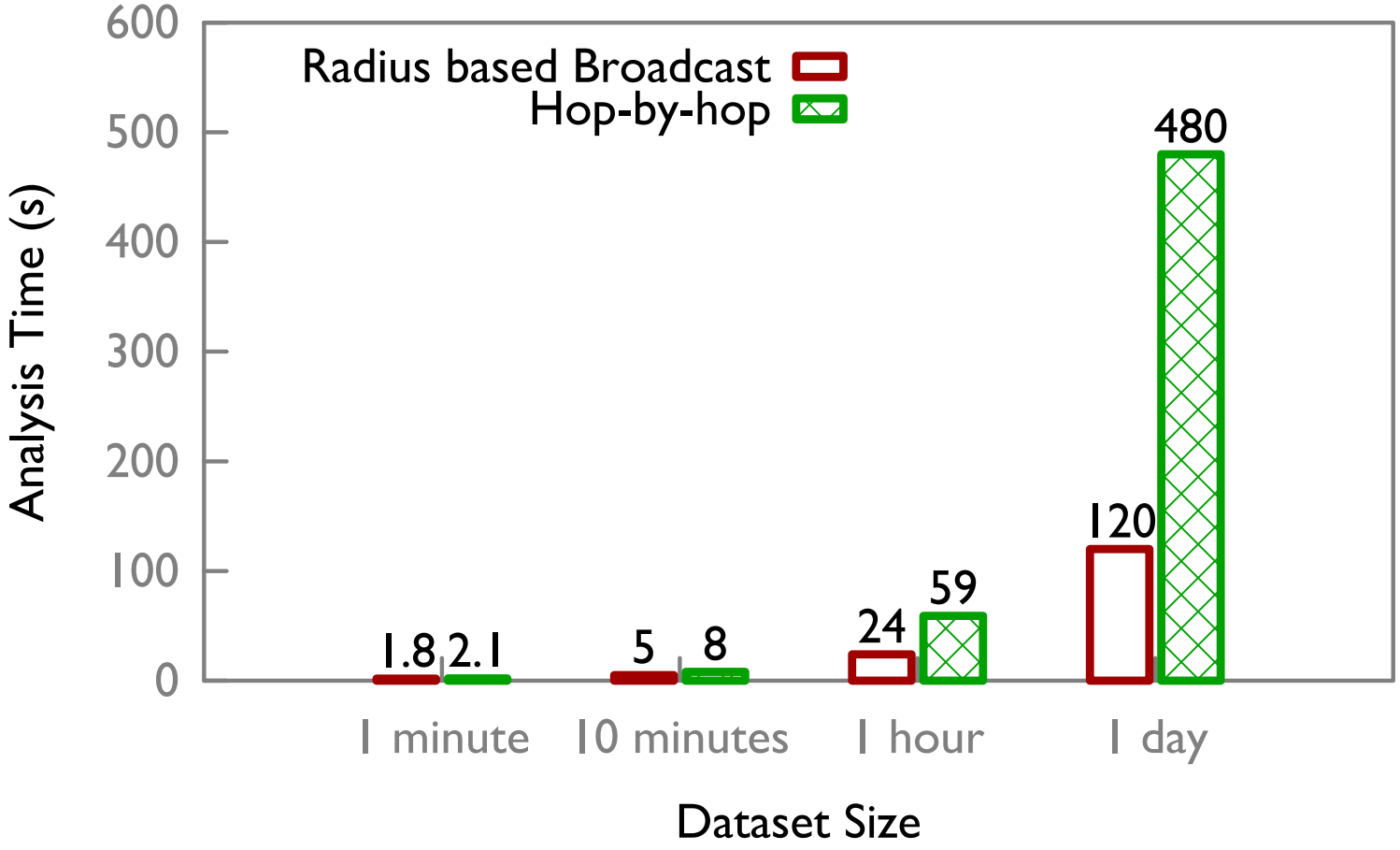
Benefits of Differential Updates



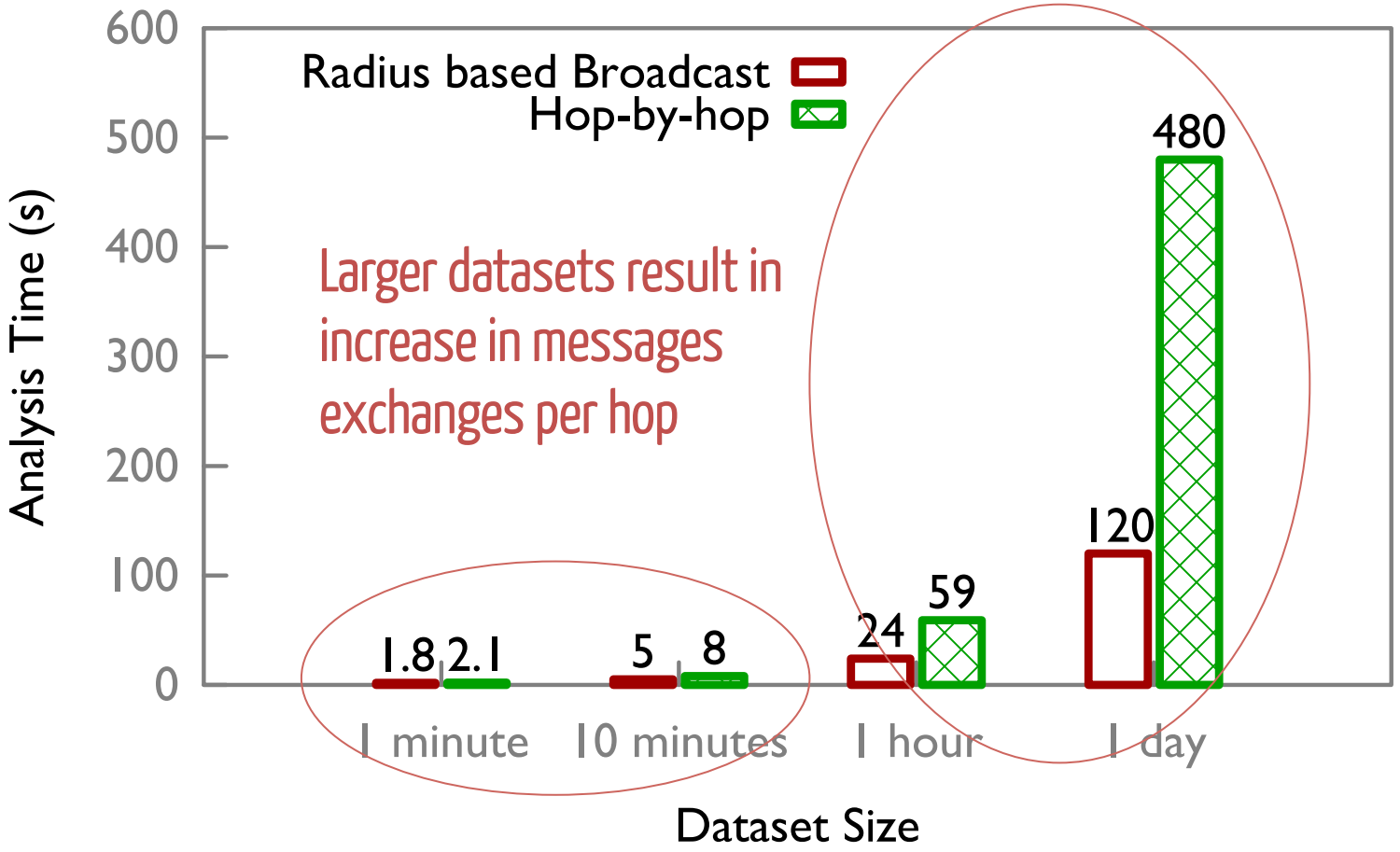
Benefits of Differential Updates



Benefits of Radius-based Broadcast



Benefits of Radius-based Broadcast



CellIQ is a cellular network analytics system that uses domain-specific optimizations to achieve 2x to 5x improvements

CelllQ is a cellular network analytics system that uses domain-specific optimizations to achieve 2x to 5x improvements

Ongoing Work:

- Using techniques in CelllQ to perform root-cause analysis on operational LTE Networks
- Generalized streaming graph analysis techniques