BUILDING A GRAPH OF ALL U.S. BUSINESSES USING SPARK TECHNOLOGIES.

Alexis Roos @alexisroos

Engineering manager / Lead Data Science Engineer

Radius Intelligence

SPARK SUMMIT EAST

DATA SCIENCE AND ENGINEERING AT SCALE

FEBRUARY 16-18, 2016 NEW YORK CITY

AGENDA

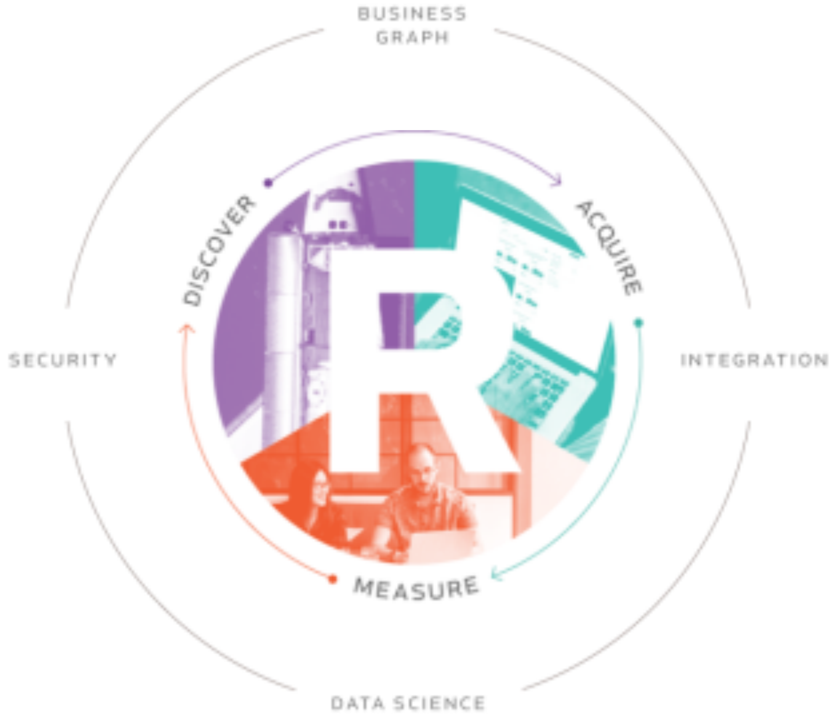
- Radius Intelligence
- Objectives and terminology
- Business graph data pipeline
- Lessons learned
- Q&As

RADIUS INTELLIGENCE

Predictive Marketing software

Radius transforms the way B2B marketers **discover** markets, **acquire** customers, and **measure** success.

Our software is powered by the *Radius Intelligence Cloud* a proprietary data science engine which provides predictive analytics, powerful segmentation, and seamless integrations.



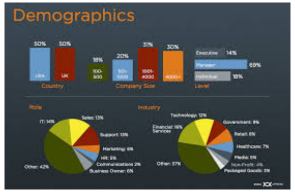
RADIUS INTELLIGENCE

- → Founded in **2009**
- → **125+ employees** (and growing)
- → Headquartered in **San Francisco, CA**
- → **$125 million** in funding to date
- → Cutting edge in **Data Science** and **Data Engineering Technologies**

STAPLES	Heartland Payment Systems
Revel SYSTEMS	AMERICAN EXPRESS
zenpayroll	Leapset
booker	main street hub

BUSINESS GRAPH OBJECTIVE

Produce an **accurate** and **comprehensive** **Business Graph** from **records** coming from dozens of sources comprising hundreds of data points such as:



...

BUSINESS GRAPH BENEFITS

Answer complex connected questions such as:

- **Employees:** Find engineering executives at a given company focusing on Big Data or security
- **News:** Show News related to companies in my marketing segment such as Funding announcements or mergers and acquisitions
- **Intent:** Show companies with a buying Intent for my product or engineering managers looking for security product
- **Organization**: Show all locations or Employees for a given business

etc

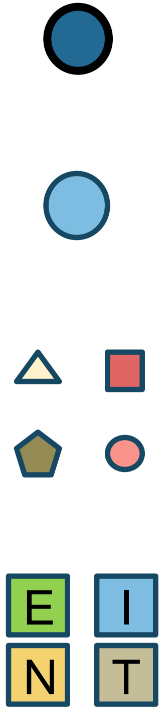
BUSINESS GRAPH TERMINOLOGY

- **Business**: a company or organization
  ex: Blue Bottle company

- **Location**: a business at a particular address
  ex: Blue Bottle at Sansome St, San Francisco

- **Attributes**: information associated with a business and/or location
  ex: address, website, phone number, etc.

- **Additional Data types:**
  ex: Employees, Intent, News, Technologies, etc.



BUSINESS GRAPH DATA PIPELINE

Data acquisition → Data preparation → Clustering → Construction

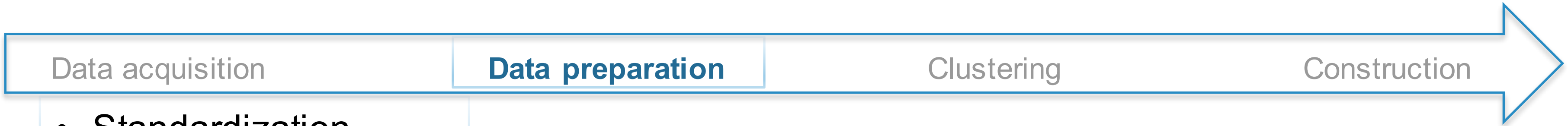
- Crawling
- Licensing

Spark
python
databricks

Dozens sources
7B+ records
50B+ Data Points

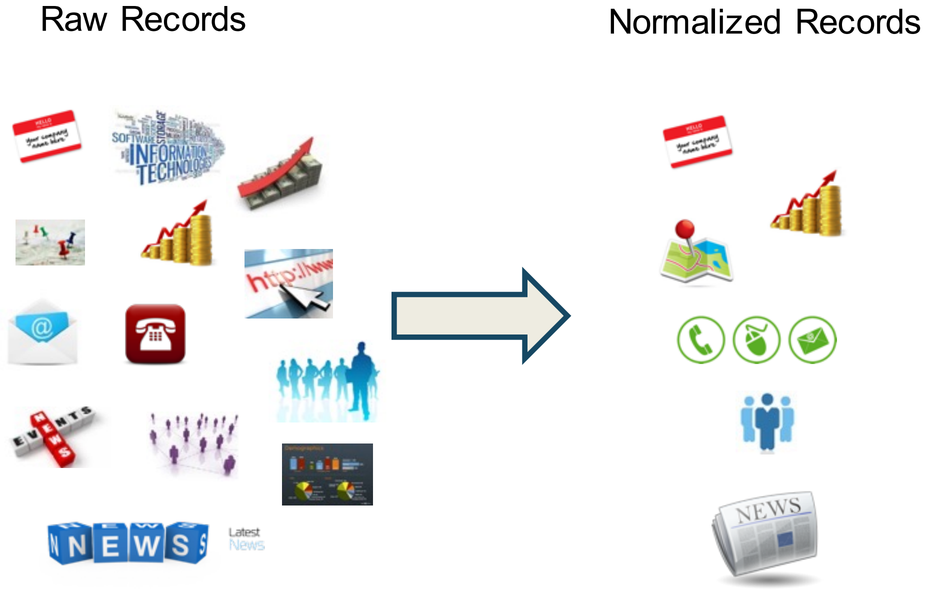
Raw Records

BUSINESS GRAPH DATA PIPELINE



- Standardization
- Validation
- Normalization

Spark
Scala
databricks



BUSINESS GRAPH DATA PIPELINE

Data acquisition → Data preparation → **Clustering** → Construction

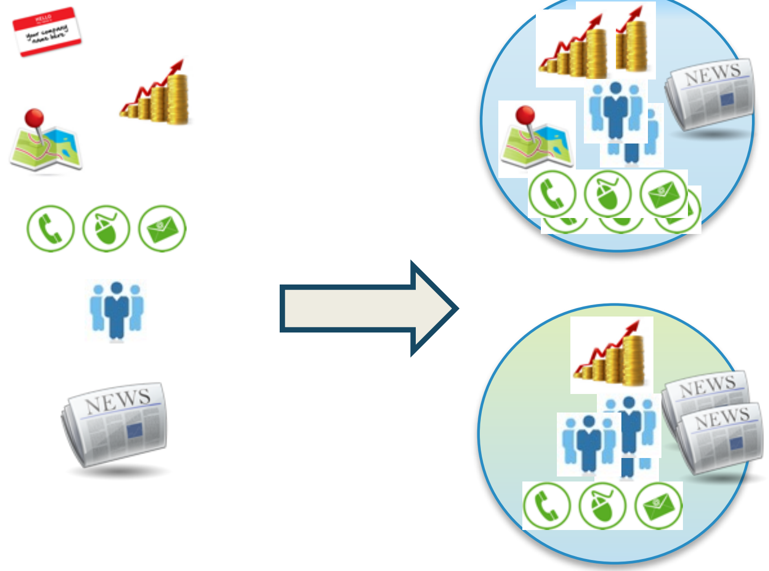
Cluster records on:

- **Address**
- Name of the business
- Websites
- Phone numbers
- Employees
- Description
- Category codes
- Etc.

Spark Scala GraphX databricks

Normalized Records

Clusters of records by address/business



BUSINESS GRAPH DATA PIPELINE

Data acquisition → Data preparation → Clustering → **Construction**

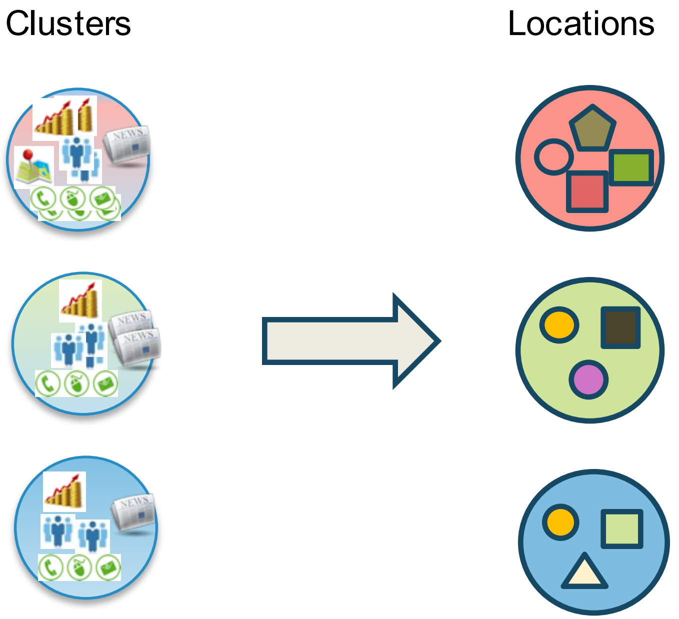
- Locations
- Businesses
& more

Construct Locations:

- Filter bad clusters
- Use all records in cluster to create Location with attributes:
  - Select best value(s)
  - Score values: confidence based on specific record features including historical
  - Impute missing values
- Etc.

Spark Scala

MLLib databricks



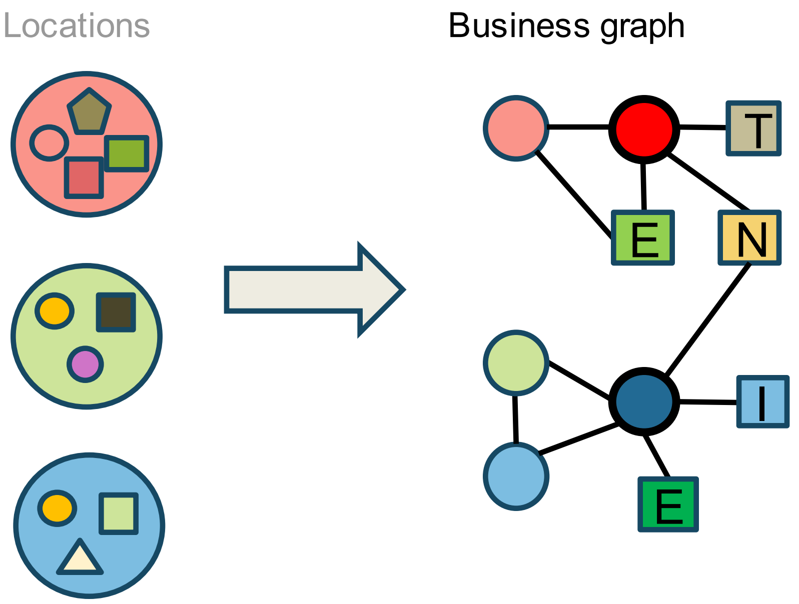
BUSINESS GRAPH DATA PIPELINE

Data acquisition → Data preparation → Clustering → **Construction Businesses & more**

Construct business & more:

- Create business and link locations
- Add business level info: news, employees, intent, etc.
- Correct/ impute fields at business level
- Etc.

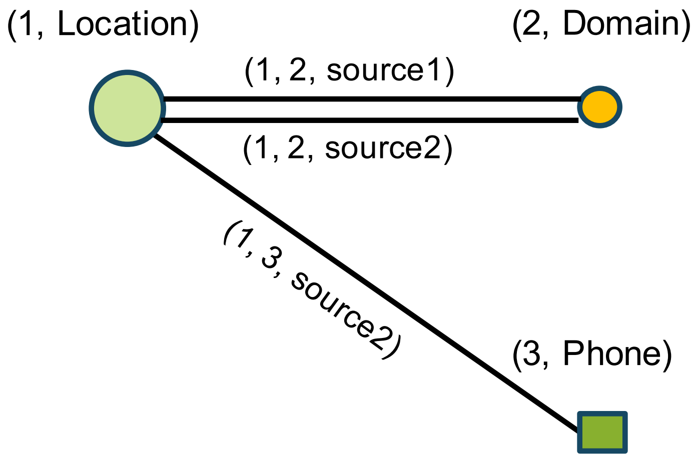
Spark Scala GraphX **MLLib** databricks



SPARK / GRAPHX

GraphX models a “property graph” which is an multi directed, attributed graph.

VD - Vertex data attribute type
ED - Edge data attribute type



```
class Graph[VD, ED] {
    val vertices: VertexRDD[VD]
    val edges: EdgeRDD[ED]
    val triplets: RDD[EdgeTriplet[VD, ED]]
 }
```

VertexRDD[VD] extends RDD[(VertexID, VD)]
EdgeRDD[ED] extends RDD[Edge[ED]]

SPARK / GRAPHX

Graph operations are like RDD operations: functional and lazily evaluated

Operations from Graph and GraphOps:

- ***Information:*** *numEdges, numVertices, inDegrees, outDegrees, degrees*
- ***Collections:*** *vertices, edges, triplets*
- ***Transformations and modifications:*** *mapVertices, mapEdges, mapTriplets, filter, reverse, subgraph, mask, groupEdges, convertToCanonicalEdges*
- ***Join:*** *joinVertices, outerJoinVertices*
- ***Aggregations:*** *collectNeighbor(Id)s, aggregateMessages, sendMsg, mergeMsg, tripletFields*
- ***Caching and partitioning:*** *persist, cache, checkpoint, unpersist(Vertices), partitionBy*
- ***Graph algorithms:*** *pregel, pageRank, connectedComponents, triangleCount*

And data operations on VertexRDD and EdgeRDD:

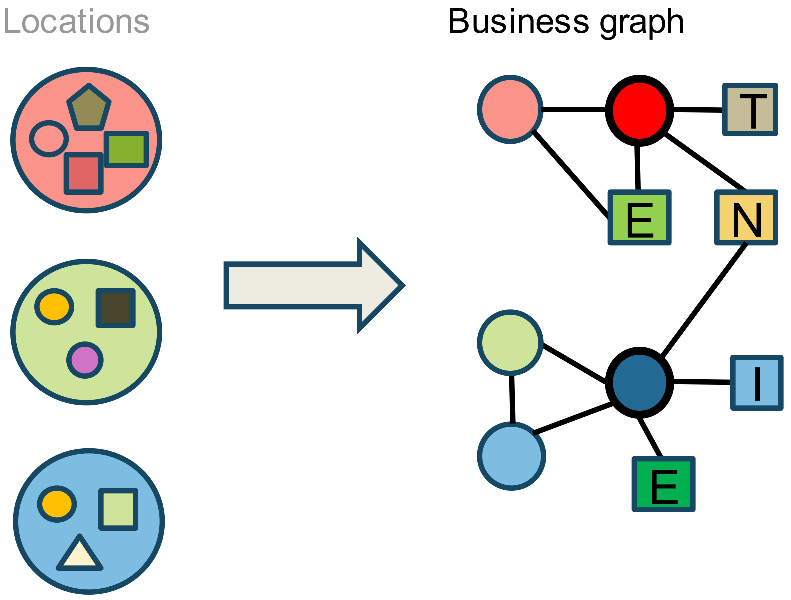
- ***VertexRDD:*** *filter, mapValues, minus, diff, innerJoin, leftJoin, aggregateUsingIndex*
- ***EdgeRDD:*** *mapValues, reverse, innerJoin*

GRAPH DATA PIPELINE CONSTRUCTION ZOOM IN

Data acquisition | Data preparation | Clustering | **Construction Businesses & more**

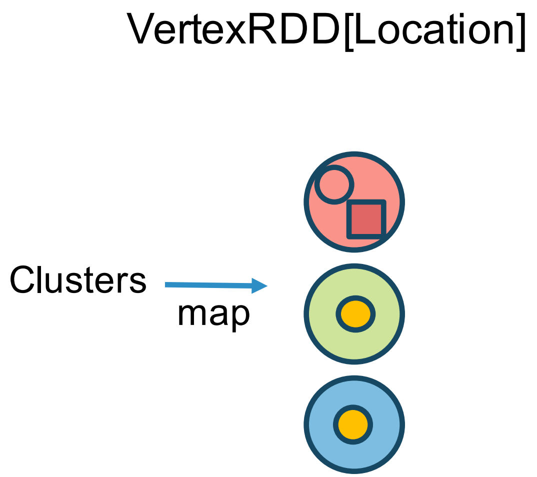
Construct business graph:

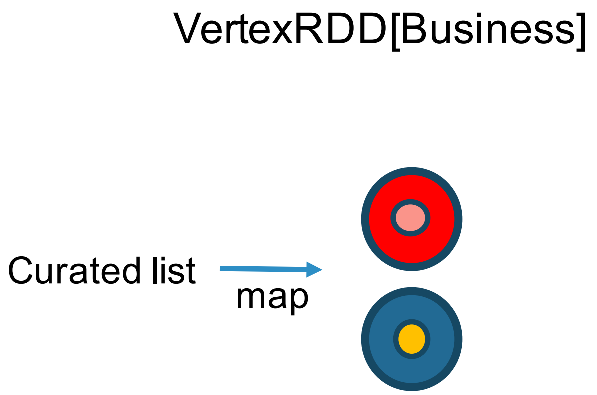
- Create business and link locations
- Add business level info: news, employees, intent, etc.
- Correct/ impute fields at business level
- Etc.



GRAPH DATA PIPELINE CONSTRUCTION ZOOM IN

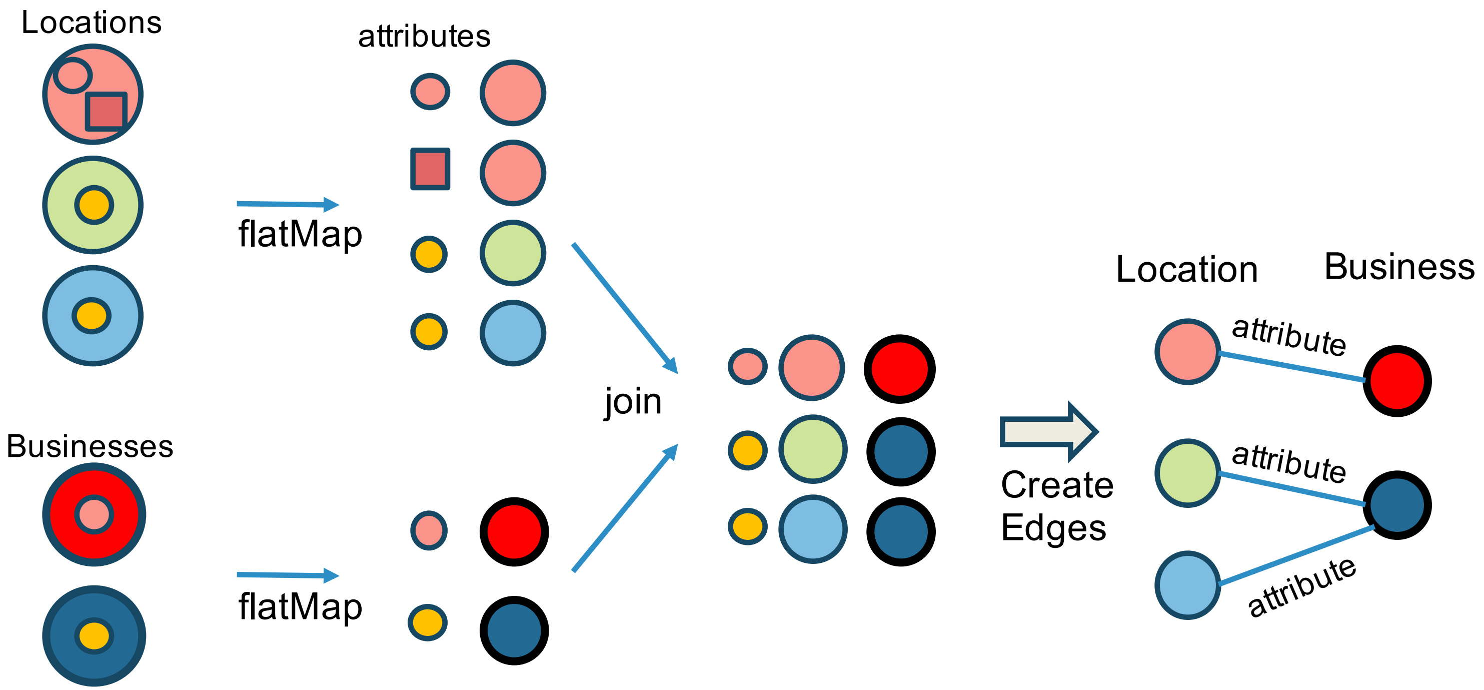
1. Create vertex for each location and each known business





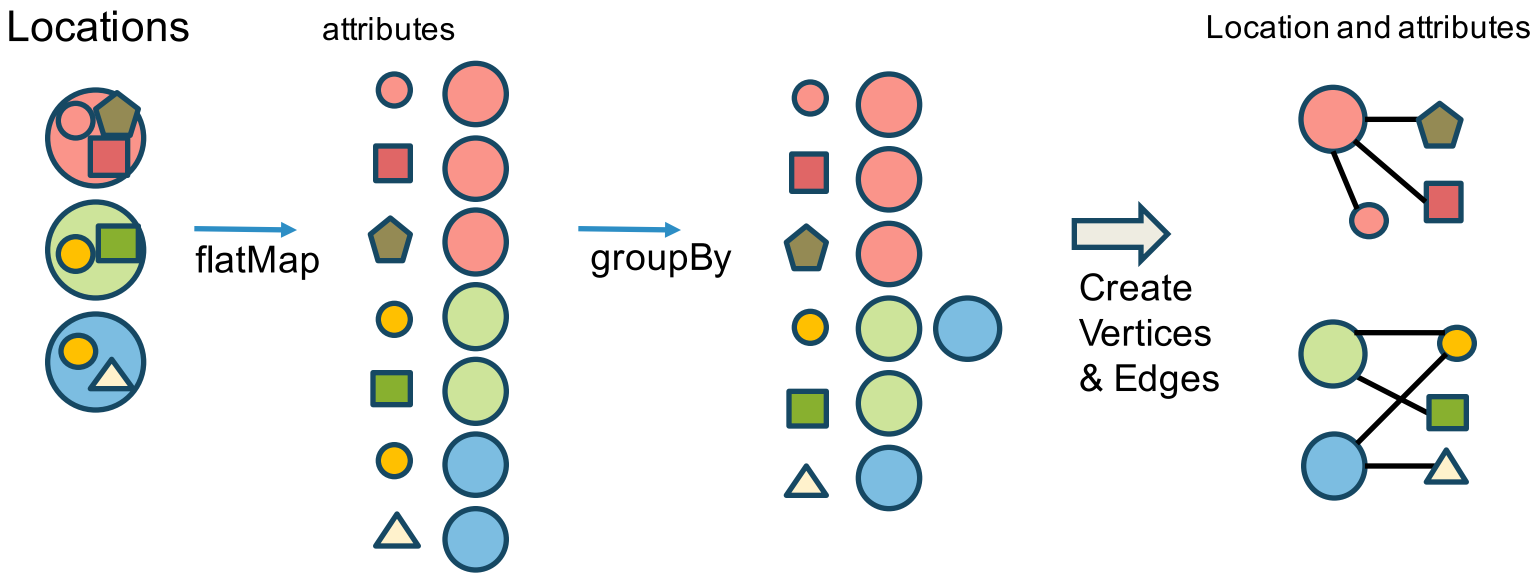
GRAPH DATA PIPELINE CONSTRUCTION ZOOM IN

2. Link Locations to Known Business per relationships (such as domain)



GRAPH DATA PIPELINE CONSTRUCTION ZOOM IN

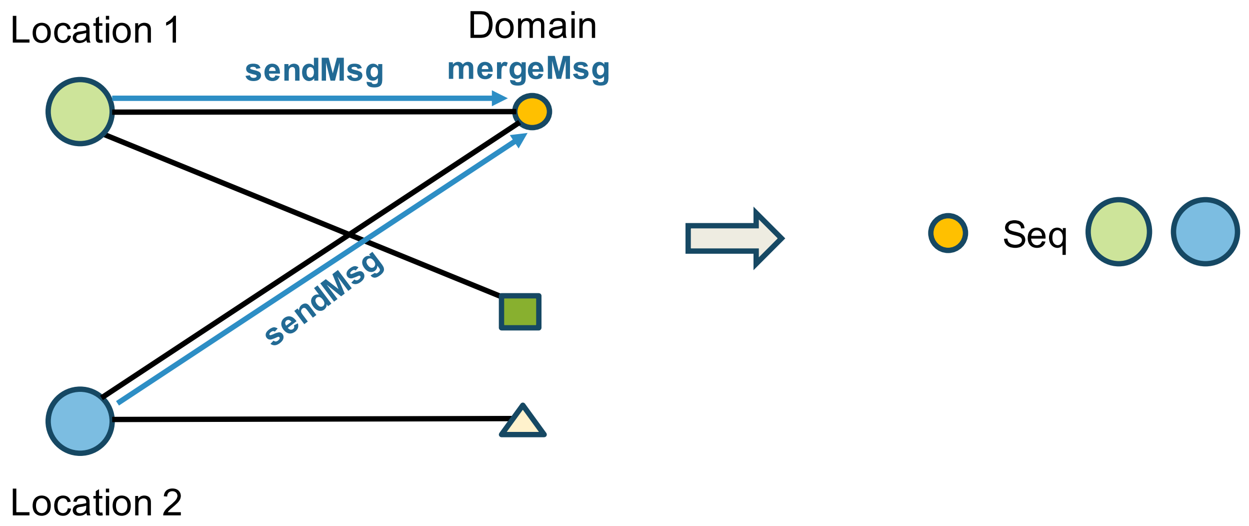
3. Create vertex for unique top attributes such as address, phone, domain, etc. and link locations to these attributes



SPARK / GRAPHX

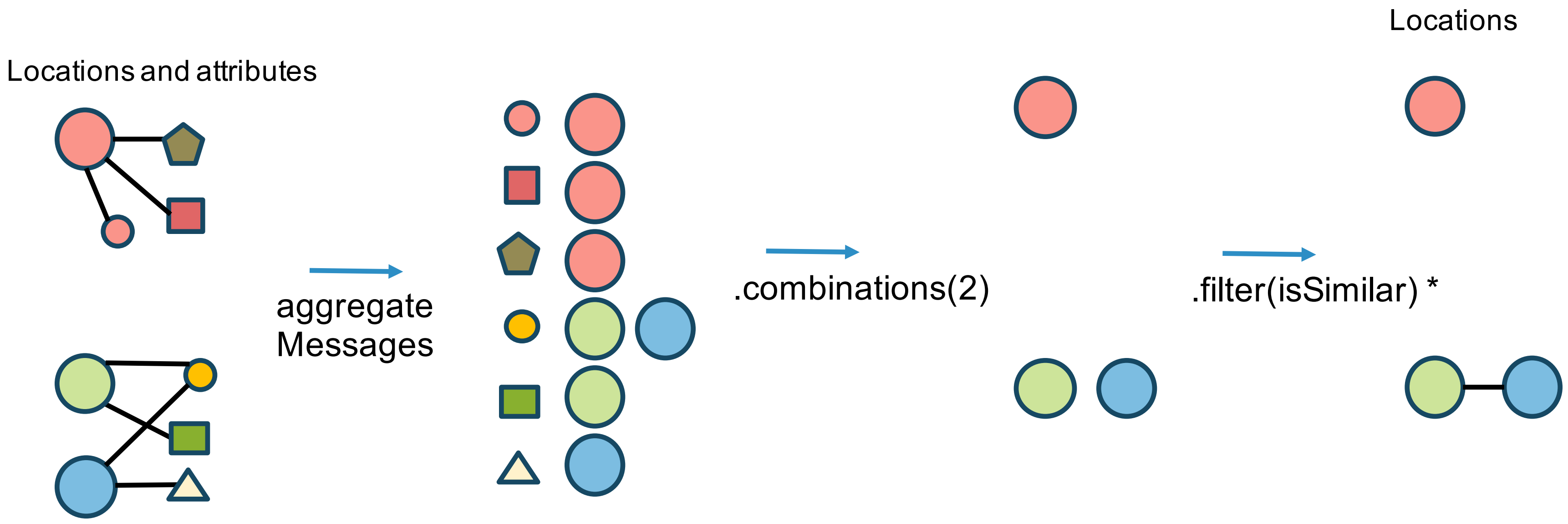
Aggregate messages:
sendMsg, mergeMsg

sendMsg: (srcID, name) **mergeMsg: x ++ y**



GRAPH DATA PIPELINE CONSTRUCTION ZOOM IN

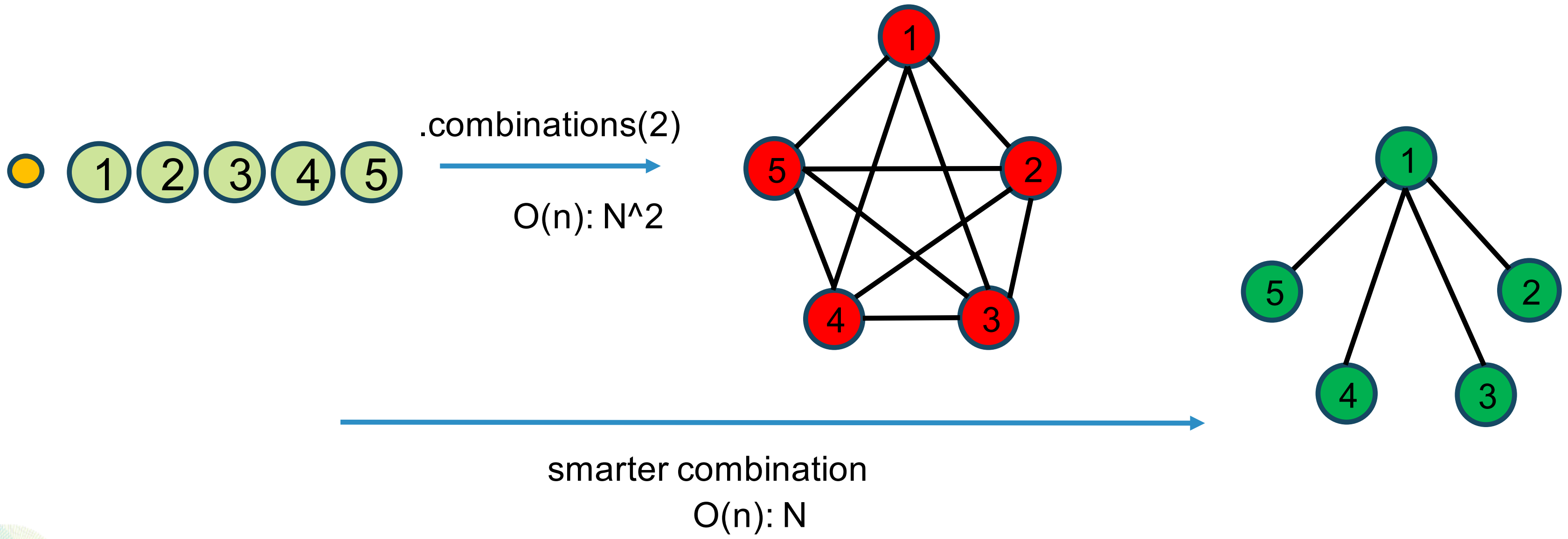
4. Create location to location edges using attributes relationship and graph cutting



* such as name matching

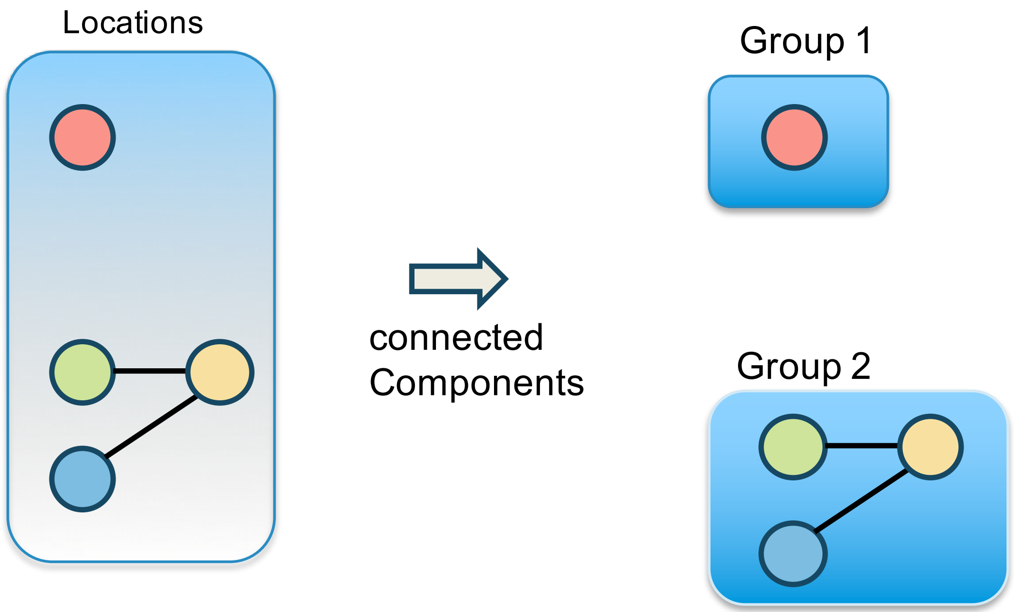
PERFORMANCE TIP

Note: .combinations (2) and bottleneck



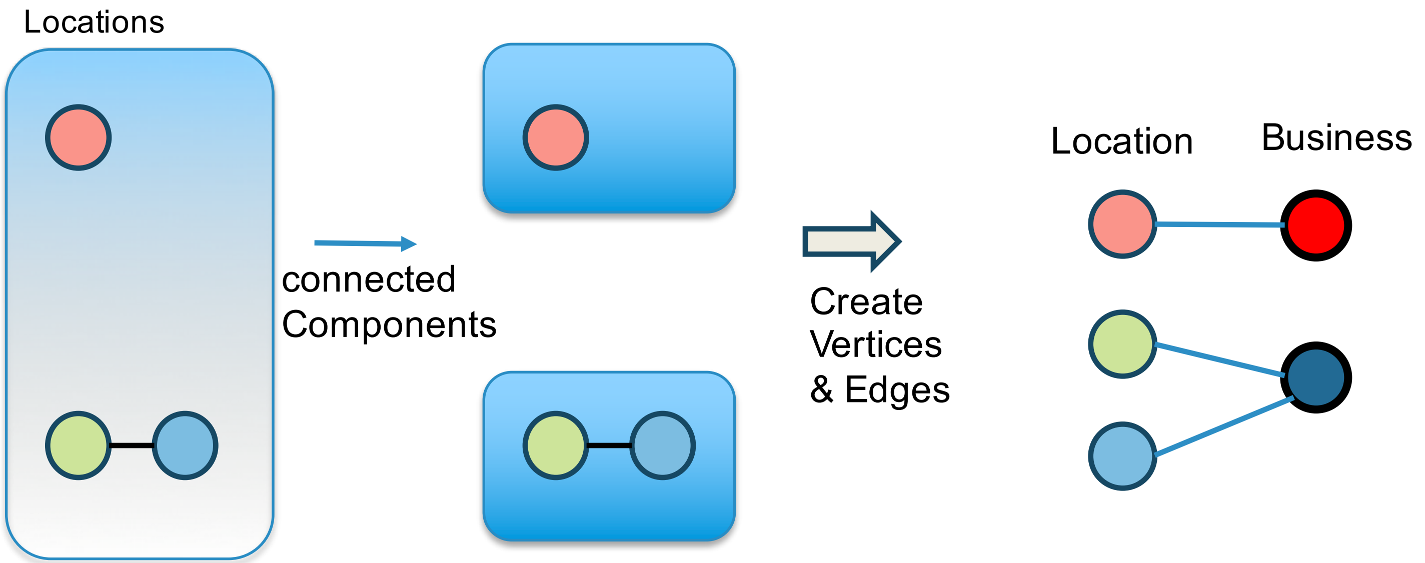
SPARK / GRAPHX

Connected components



GRAPH DATA PIPELINE CONSTRUCTION ZOOM IN

5. Call connected components on linked locations and create businesses



PERFORMANCE TIP

GraphX Connected components implementation already performs optimizations:

- Incremental Updates to Mirror Caches
- Join Elimination
- Index Scanning for Active Sets
- Local Vertex and Edge Indices
- Index and Routing Table Reuse

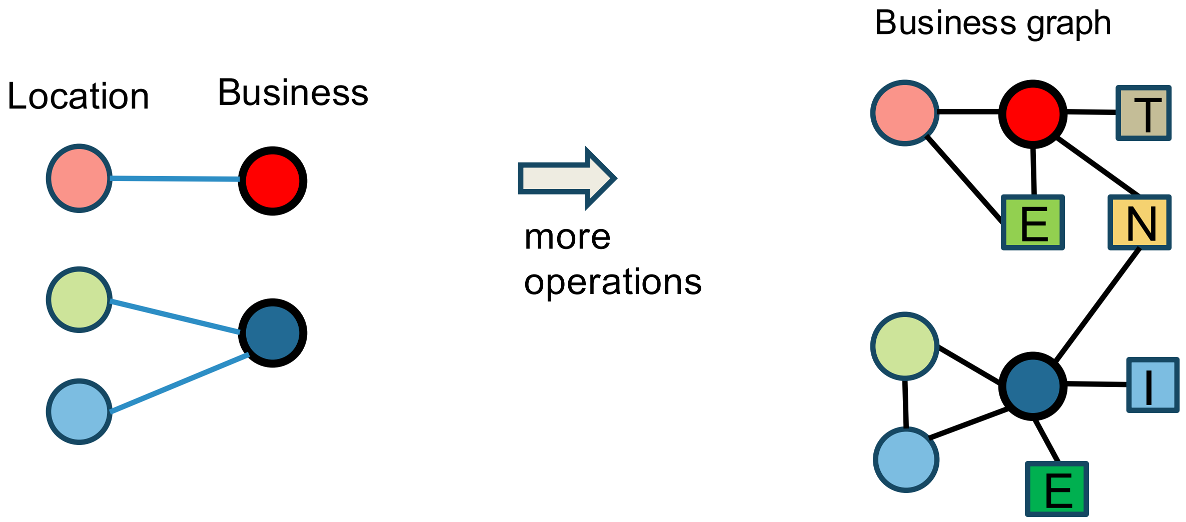
But algorithm is iterative so optimization matters:

- Vertices and edges should be in memory: more efficient to subset / keep the graph minimal or filter and rejoin later
- Check-pointing improves performance (less lineage)

GRAPH DATA PIPELINE CONSTRUCTION ZOOM IN

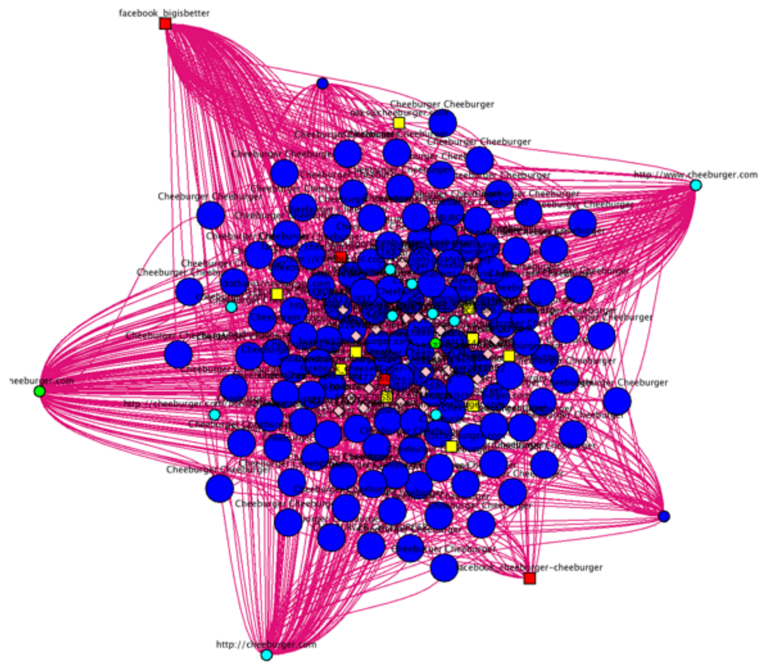
6. More steps:

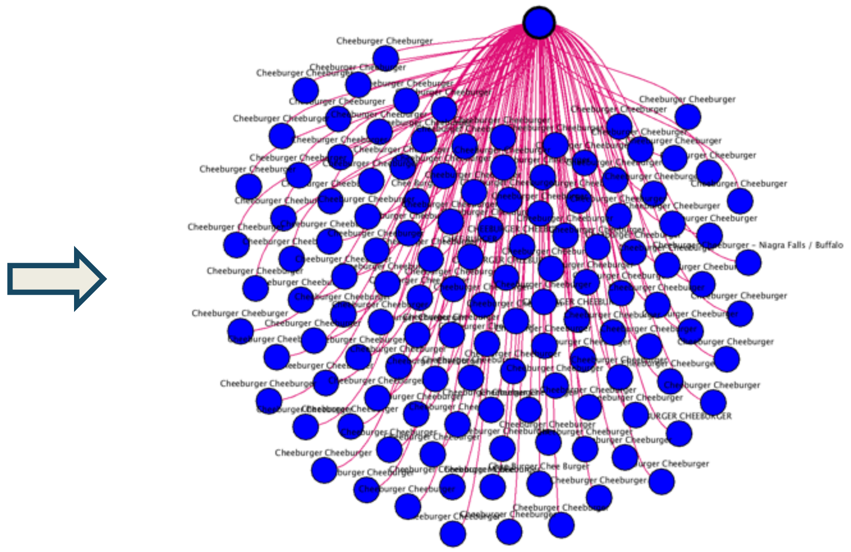
- Assign headquarter
- Join additional information: Employees, News, Intent, etc.
- Further corrections / imputations: attributes, revenues, etc.



GRAPHICAL VIEW

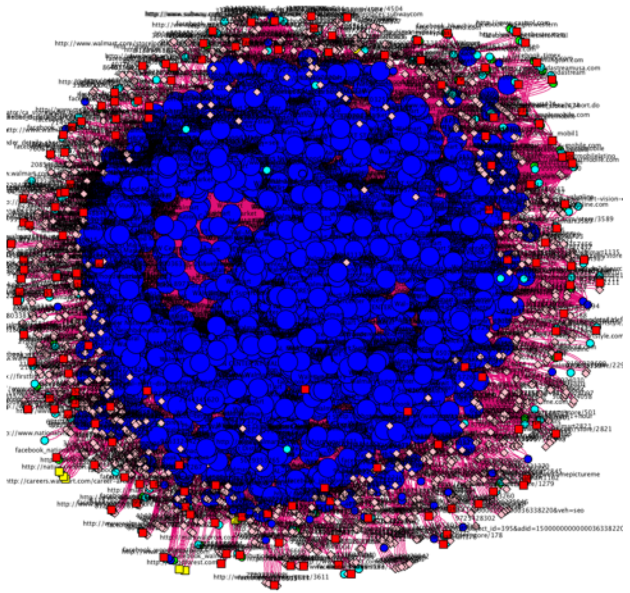
From Locations with attributes to Locations with Business

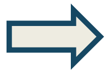


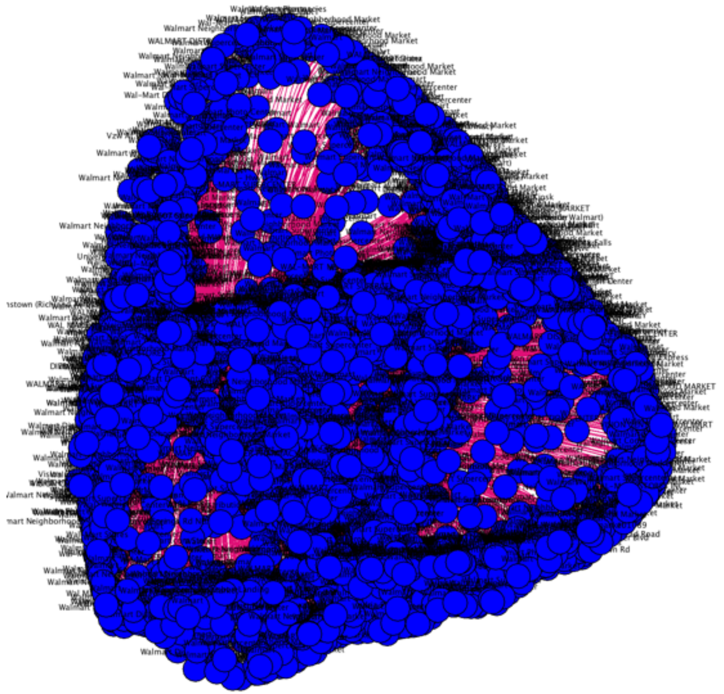


GRAPHICAL VIEW

From Locations with attributes to Locations with Business







LESSONS LEARNED

- Using a Graph allows us to naturally model business relationships and append new data types over time. Also allowed us to debug data!
- Creating and updating the graph is easy using RDD operations
- GraphX scales: 250M vertices and 1b edges in clustering
- Edge cutting is an art as much as a science
- Connected Components is an expensive operation

THANK YOU.

WE ARE HIRING!
http://radius.com/jobs/

Alexis Roos
alexis@radius.com
@alexisroos