ACME GOURMET MEALS

Tech Stack Modernization

W205 - Fall 2022 - Reid - Section 3
Tigran Poladian, Sid Gupte and Kumar Kallurupalli

Introduction

- Introduce new tech stacks
  - MongoDB
  - Redis
  - Neo4j
- Provide Use Cases for business improvement
- Demonstrate the use of NoSQL

MongoDB

- Document based scalable storage mechanism for unstructured data
- JSON format – existing web applications can access data
- faster access and display of data over traditional RDBMS
- Rail operators tend to use **document** schedule formats
    - collaboration with the BART system
    - BART timetables and transportation routes
    - City transit trains change daily schedules often

- MongoDB ideal to store planned and last minute schedule and route changes
    - import the new data as a self contained document
    - RDBMS, the schedule and route files would first need to be parsed, data loaded into appropriate columns and possibly table or tables to be re-indexed which are costly operations.

Redis

- Redis is a NoSQL database, in-memory key-value storage for
  - extremely fast read and write operations
  - real-time data streams
- When compared to traditional RDBMS databases that rely on secondary memory, Redis provides significantly faster reads and updates
- Redis has limits involving the storage of large files or binary data.
  - disadvantage compared to other NoSQL databases like MongoDB
  - primary memory is significantly more expensive than secondary memory which adds cost as a factor when considering a switch to Redis.
  - However, the overwhelming speed with which you can look-up and update data make Redis a clear favorite when real-time data is a part of the business use-case.

Neo4j

- Recommend using BART system for deliveries.
- Local road routes from Train stations to customer addresses.
- BART Lines to be captured as graphs
- Stations as nodes, lines as relations
- Need to run shortest path algorithm to suggest best delivery route.

- Neo4j would be ideal for holding the BART system in the form or a graph.
- Provide out of the box implementation of shortest path and other algorithms.
- RDBMS will not be able to capture the BART system as a graph and any shortest path algorithms will have to be implemented by the Data Engineering Team.

Business Improvements

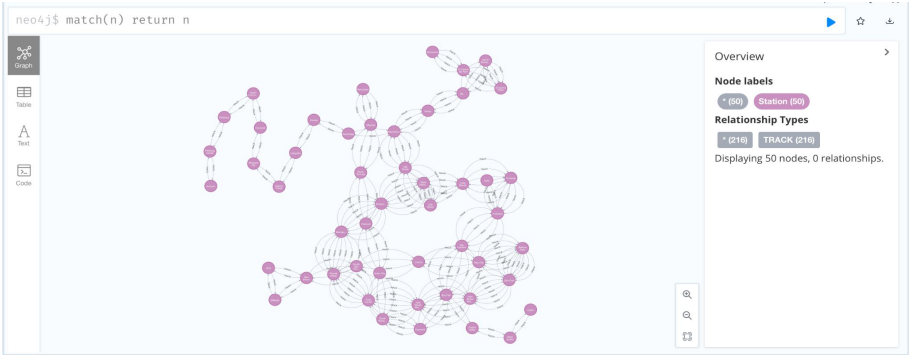
- Recommend BART system for deliveries to decongest traffic
- Delivery trucks / robots to run local routes from each train station
- Identify current patterns to discover opportunities
- Offer convenient Station pickup / drop off or mobile delivery
- Offer seasonal popup locations

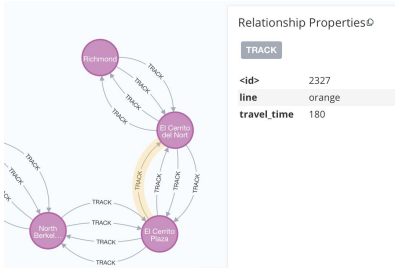
IMPROVE EXECUTION

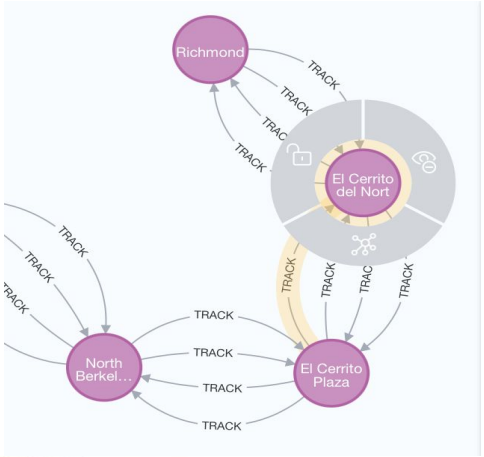
- Redundant system design – multiple channels
  - Re-route transport of goods
  - Utilize mobile robots to bypass congestion
  - Last mile solution
- Dynamics of customer flow
  - pop-up pick up and drop off locations
  - Route mobile robots to congested stations
  - adjust quickly to changed routes and schedules and ensure that our product deliveries are on time

Neo4j Graph

- Graph layout - BART stations
- Relationships - tracks
- Nodes contain
  - Location
  - Active customers near station per station zip code
- Tracks show transit time and Line color







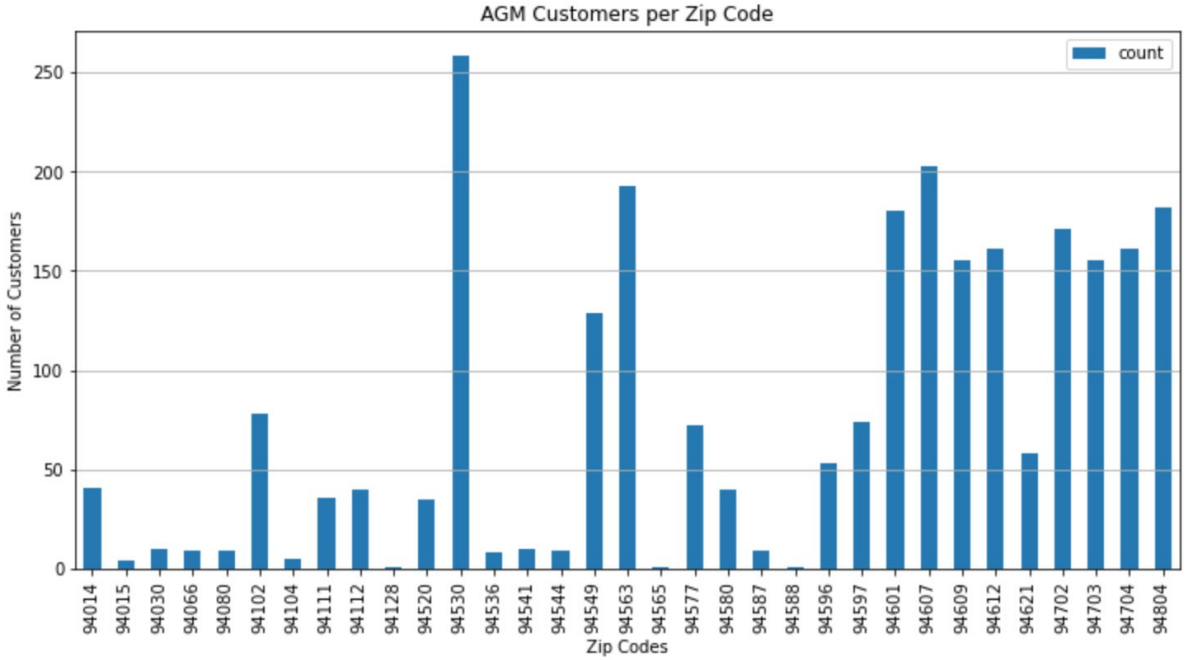
Node Properties

Station

<id>	51
latitude	37.925183
longitude	-122.316939
name	El Cerrito del Norte
pop	258
transfer_time	311
zip	94530

Graph Data Science Results

- Analyze customer data
- Identify large clusters of customers by zip = Station



Graph Data Science

- Cluster analysis and graph analysis
  - Page Rank algorithm using Stations as Pages to determine most used stations
  - Mapped to customer counts per area
- Determine best locations for permanent pick up and drop off locations
- Track patterns and determine pop up locations
- **Identify high rank scores with low customers to drive business in those areas**
  - **West Oakland, San Bruno**

	name	score	customers
0	Coliseum	1.387533	58
1	Bay Fair	1.375688	40
2	MacArthur	1.291159	155
3	San Leandro	1.230845	72
4	Balboa Park	1.163400	40
5	West Oakland	1.158007	0
6	San Bruno	1.142995	9
7	19th Street	1.117343	161
8	Pittsburg Center	1.115549	1
9	Milpitas	1.115419	0
10	El Cerrito del Norte	1.110790	258
11	Glen Park	1.071030	0
12	Embarcadero	1.067680	36
13	Fruitvale	1.065852	180
14	Pittsburg	1.043628	0
15	Warm Springs	1.043566	0
16	El Cerrito Plaza	1.036565	0
17	24th Street Mission	1.022083	0
18	Montgomery Street	1.019887	0
19	Fremont	1.003626	8
20	12th Street	1.003171	203
21	North Concord	1.002297	0
22	Daly City	0.999236	4
23	16th Street Mission	0.998272	0
24	Powell Street	0.997340	5
25	South San Francisco	0.991943	9
26	Civic Center	0.990911	78
27	North Berkeley	0.990261	171
28	Lake Merritt	0.984534	0
29	Union City	0.977743	9
30	Concord	0.974632	35
31	South Hayward	0.960322	9
32	Downtown Berkeley	0.953952	161

Graph Data Science

- Determine shortest path from a pick up to drop off location
- Algorithms on graph data
  - Shortest path using transit time
  - A* based on location
  - Identify other business relevant measures
- Mobile path planning

Image from berkeley news

Antioch to Richmond shortest path

```
------------------------------
   Total Cost:  4380
   Minutes:  73.0
------------------------------
Antioch, 0, 0
Pittsburg Center, 420, 420
Pittsburg, 600, 1020
North Concord, 360, 1380
Concord, 180, 1560
Pleasant Hill, 360, 1920
Walnut Creek, 120, 2040
Lafayette, 300, 2340
Orinda, 300, 2640
Rockridge, 300, 2940
MacArthur, 240, 3180
Ashby, 240, 3420
Downtown Berkeley, 180, 3600
North Berkeley, 120, 3720
El Cerrito Plaza, 180, 3900
El Cerrito del Norte, 180, 4080
Richmond, 300, 4380
```

Redis Use Cases & Implementation

- For AGM, through the use of Redis:
  - track delivery schedule, significant traffic events on delivery routes and the inventory at each of our warehouse locations
  - Accurate delivery times for customers
  - re-route deliveries traffic congestion or road closures
  - With a cargo of perishable food items, speed is paramount
- In terms of implementation, we would have key-value pairs that represent warehouses paired with locations, inventory and the location of the closest warehouse (the closest warehouse can either be updated real-time based on whether there are situations where traffic can affect the proximity in terms of travel time or computed offline).
- Similarly, we would have key-value pairs that represent locations of restaurants or other delivery locations, purchased inventory and the location of the closest warehouse.
- Finally, a key-value pair that stores and updates road segments involved in our delivery route and we will update those with live traffic incidents.
- We can then use the combination of all this data to update our delivery plans to reroute our vehicles and revise our estimates. If we are set to be significantly earlier than our estimate, we can even add additional deliveries to the route

Path Planning

Integrate with Google Maps to task mobile delivery units for last mile and to bypass track closures

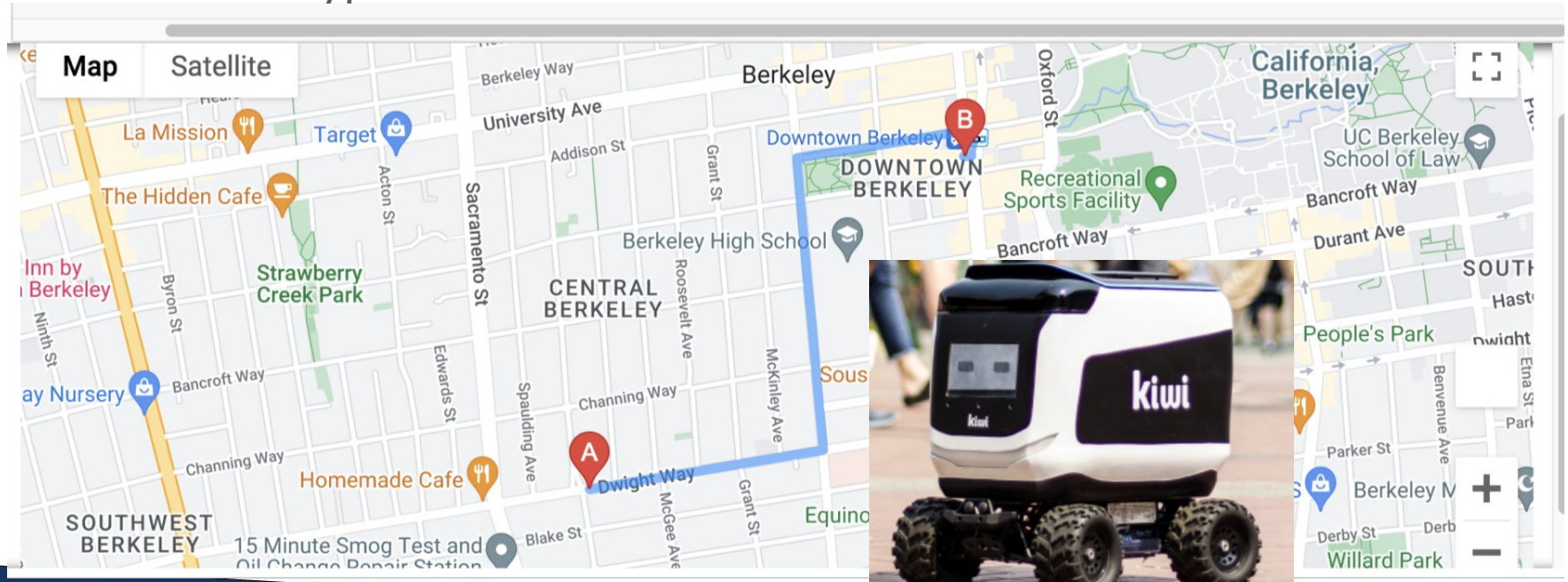


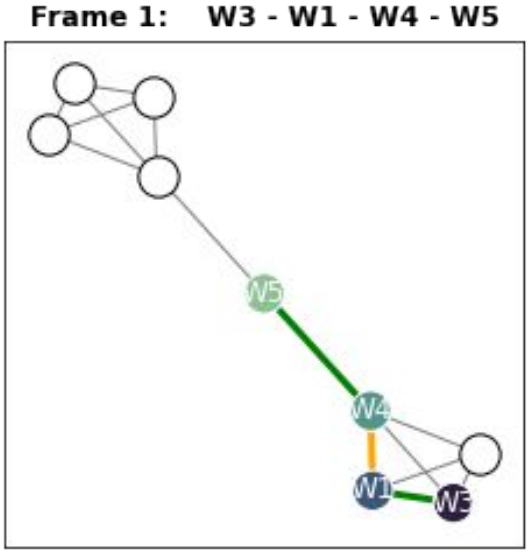
Image from kiwi

Use Cases – Redis Traffic Flow

- Unpredictable traffic makes delivery estimates unreliable
- Sudden traffic events usually cause the most significant delays
- Batch updates to our graph are too slow
- For more complex routes, pulling data causes a delay too

Use Cases – Redis Traffic Flow

- Traffic Information is available from several streaming data sources
- Route streaming data to Redis for our primary routes to and from warehouses
- Pull latest traffic data from Redis cache and re-calculate shortest path
- At our scale, Redis provides a sizeable benefit with a minimal cost



Other Enhancements

- Presented Use Cases are based on customer zip codes compared to station locations
- Models can easily extend to using customer specific addresses
- Integrate Redis real time updates with MongoDB schedule updates and Neo4j graph


QUESTIONS?