Geographic Resource Dispatch

a tool to facilitate optimization of distributed computing resources based on real-time market signals

INSIGHT

The Problem

Businesses increasingly rely on leveraging large data sets and distributed computing resources for processing.

Geographic distribution of these resources leads to variations in characteristics and capacity of the nodes that can be leveraged to optimize output.

Inherent latency and reliability of network communications presents a challenge to reliable determination of the optimum dispatch scenario.

The Solution

A prioritized job queue, with metadata on the expected compute-hours and urgency for completion.

Real-time data streams (weather, energy cost) and stream processing (cooling energy) to determine resource capacity and pricing per compute-hour.

Optimization algorithm to generate dispatch queue for submitting jobs across the geographically distributed resource nodes.

Monitoring and visualization to facilitate manual interaction and analysis.

The Demo

github.com/simchuck/InsightDataEngineering/GeographicResourceDispatch

The Approach

Streaming architecture (Kafka) and stream processing (KSQL) to accommodate real-time ingestion and processing use case.

Simulated data streams (Python) from static store (S3) of representative data to facilitate development for maximum flexibility.

- Streaming interval
- Geographic distribution
- Network performance issues

Python implementation to optimize available development time.

The Challenge(s)

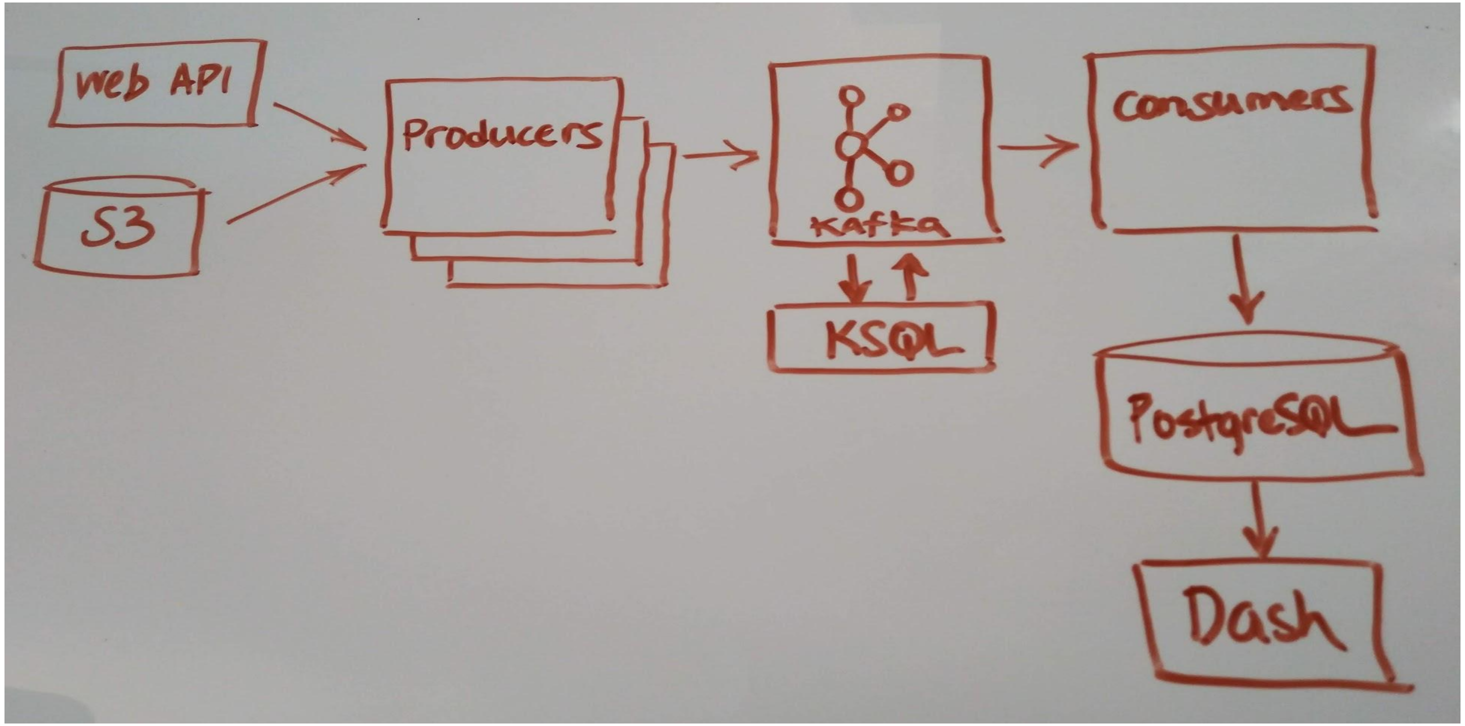
Scale

- Rate of ingestion and processing
- Number of resource nodes
- Geographic distribution of nodes

Timing

- Coordination across nodes via source timestamp
- “Expired” events

The Technology



The Future

- Scaling (more resource nodes, shorter time intervals)
- Time coordination of distributed streams (latency, packet loss)
- Use of AWS Lambda (stream filtering and transforms, API calls)
- Parallelizing API calls with Apache Airflow
- Improved architecture (Connect API, move logic to KSQL, database)
- Metrics and logging (Prometheus, Grafana)
- Refactor to use native Java APIs for more performance
- Additional streams and logic for forecasting

The Fellow

Charles Simchick

charles.simchick@gmail.com
linkedin.com/in/csimchick
github.com/simchuck/

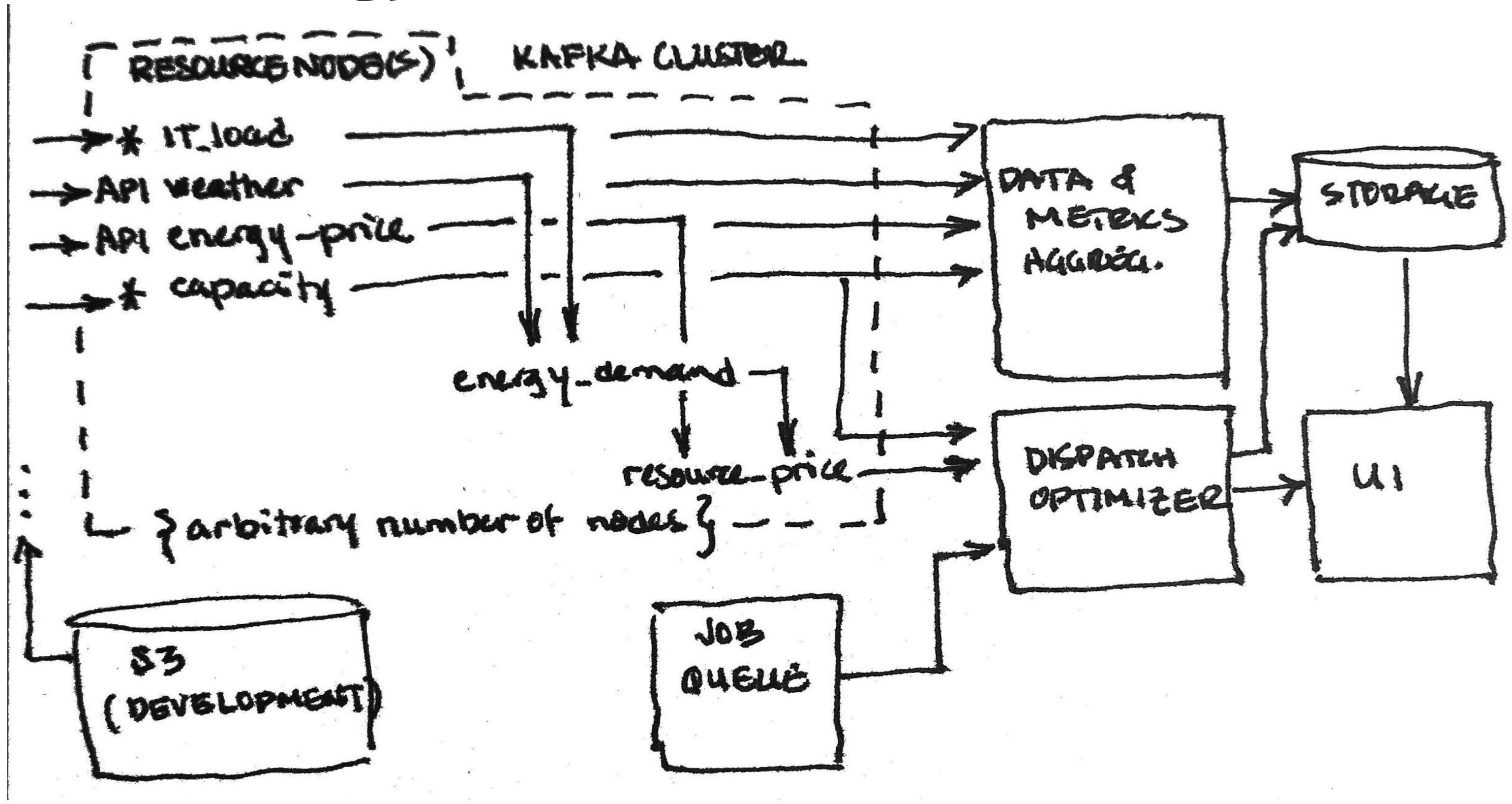
Mechanical Engineer
Energy & Sustainability
Data Advocate

INSIGHT

The Backup Slides

Coming soon...

The Technology


RESOURCE NODE(S)
KAFKA CLUSTER
* IT_load
API weather
API energy-price
* capacity
energy_demand
resource_price
{arbitrary number of nodes}
DATA & METRICS AGGREG.
STORAGE
DISPATCH OPTIMIZER
UI
JOB QUEUE
S3 (DEVELOPMENT)