Learning-Based Creation of Data Mesh Architectures


Photo courtesy of Martin Jernberg on Unsplash
Meet Tim the Beaver!

- Tim the Beaver is MIT’s mascot
  - Beavers are resourceful and capable engineers!
- So one day, a startup asks him for help in building their cloud data infrastructure
  - … and he turns to our community for some advice
The modern cloud data mesh

AWS Aurora
(OLTP DBMS)

AWS S3
(Object Storage)

AWS EMR
(Cluster Processing)

Spark
Presto
Hive

Snowflake
(Data Warehouse)

Google BigQuery Omni
(Data Lake Query Engine)
Tim the Beaver’s job sucks!

- Pick an **OLTP engine** (Aurora, PostgreSQL, MySQL?)
- Pick a **data warehouse** (Redshift, BigQuery, Snowflake?)
- Set up **ETL pipelines** among engines
- Add **NoSQL engines** for caching or other data formats?
- Configure a **data lake execution engine** (Spark, Presto?)
- Tell users which data system(s) to use for their needs!
“One size does not fit all” leads to specialization...

...but specialization leads to a paradox of choice!
Picking the right system(s) can matter a lot

- It depends on the workload!
- Aurora/Redshift (2 vCPUs, ~16 GB memory)
- Two queries with the **same join template** adapted from the Join-Order benchmark
- Query A2 is faster on Aurora due to **selectivity** and the **presence of indexes**
- Running all analytical queries on Redshift is not always the best choice!
Can we make Tim the Beaver’s job easier?

- **Too many choices** for Tim the Beaver and others in his shoes!
- Users often do not care about the specific choices
  - They want something that works well enough
  - They also want **less complexity**!
- **The holy grail**: One system with state-of-the-art performance for every possible workload
- Without discarding **existing** specialized systems!
Introducing BRAD: “One interface fits most”

BRAD abstracts away a mesh of specialized cloud database engines as “one system”

BRAD autonomously manages the data mesh: adding/removing/scaling engines as needed

Users interact with BRAD using SQL

BRAD picks the best engine(s) for the workload.
But what about prior work on polystores?

- **BRAD** is a polystore, but the **problem domain** is different

- BRAD tackles infrastructure management in the **modern cloud**

- BRAD leverages **learned models and policies** to manage the data mesh

- BRAD operates on **one data model** (relational), making cross-engine optimizations easier
Multi-engine query execution for the best of N worlds

- BRAD finds the best way to execute a query:
  - Entirely on one engine
  - In parts across multiple engines
Data is where you need it when you need it

- BRAD finds the best way to execute a query:
  - Entirely on one engine
  - In parts across **multiple engines**

- BRAD handles **data placement** and **movement** across engines
Forget about mesh management… BRAD does it too!

- BRAD finds the **best way** to execute a query:
  - Entirely on one engine
  - In parts across **multiple engines**
- BRAD handles **data placement** and **movement** across engines
- BRAD makes engine addition/removal/scaling decisions based on **workload forecasts** and observed loads
A sneak peek on how we’re getting there!

- 🚀 Use learned cost models to predict the best engine for a query

- ☑️ Optimize the data mesh deployment for cost/performance using offline system experiments and cost models

- ☑️ Forecast the future workload to anticipate when to re-plan the data mesh deployment
There are many exciting challenges ahead

- ✂ How should we **split** and **optimize** plans across engines?

- 📚 How do we **collect** and **reconcile** performance statistics from **different** database engines to make global service provisioning decisions?

- 🔍 Can we **automatically discover** the performance characteristics of a cloud database service and autonomously decide to bring it into an existing mesh?

- 🤖 How do we **effectively** involve a **human-in-the-loop** to make **application-specific** decisions?

- 🌍 How to effectively **interoperate** with existing data meshes?
BRAD simplifies data mesh creation and management

- One unified interface over a mesh of specialized cloud database services
- Learned models and policies that automate data mesh use and management
- Pay only for what you use and automatically get state-of-the-art performance
- Tim the Beaver can get back to his mascot duties!

Do you have a cloud data use case that could benefit from this work?

Please get in touch: iohtap@csail.mit.edu