2592000

Development of a Cloud-based Data Match-Up Service (CDMS) in Support of Ocean Science and Applications

Nga Chung[1], Thomas Huang[1], Vardis M. Tsontos[1], Stepheny Perez[1], Wai Phyo[1], Shawn R. Smith[2], Homer McMillan[2], Thomas Cram[3] Zaihua Ji[3], Kimberly Sparling[4]

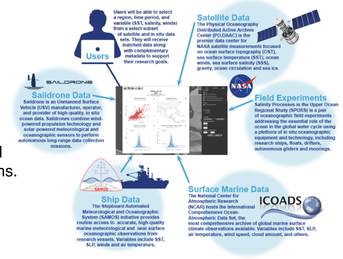
[1] Jet Propulsion Laboratory, California Institute of Technology [2] Center for Ocean-Atmospheric Prediction Studies, Florida State University

[3] National Center for Atmospheric Research [4] Saildrone

Jet Propulsion Laboratory
California Institute of Technology

Overview

The Cloud-based Data Match-Up Service (CDMS) is a collaborative effort between NASA JPL, COAPS, NCAR, and Saildrone. CDMS is an extension of the Distributed Oceanographic Match-Up Service (DOMS) which was funded by the NASA AIST program. CDMS will provide a mechanism for users to input a series of geospatial references for satellite observations and receive the in situ or satellite observations that are matched to the primary satellite data within selectable temporal and spatial search domains.



The software stack that enables CDMS match-up capability is available via the Apache Science Data Analytics Platform (SDAP), which is an Apache incubator project. Under the ACCESS program, the team plans to deliver a production-ready match-up capability that fully leverages cloud-native services.

Why is CDMS Needed?

CDMS provides a generalized match-up capability that can be used for:

- Calibration/validation
- Mission data processing
- Science use cases that require colocated data retrieval

CDMS eliminates the need to house satellite and in-situ data on one's local computer while computing one-off match-up.

CDMS provides reproducibility so other team members can execute the same request and get the same result.

CDMS allows for a fully customizable match-up request. Users can specify parameters that allow their request to match their exact needs. Spatial and temporal tolerances can be modified on the fly to customize match-up results.

CDMS is extensible. It has been designed such that additional data providers can be added.

CDMS is flexible and allows the following:

- Satellite (any level) to in-situ collocation
- L2 to L3/L4 satellite collocation
- L3/L4 to L2 satellite collocation

Architecture

In situ data

CDMS infuses common data access services in AWS that are managed by the participating institutions:

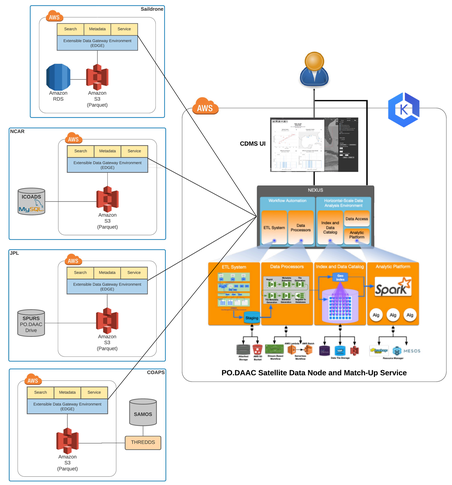
- JPL (SPURS)
- COAPS/FSU (SAMOS)
- NCAR (ICOADS)
- Saildrone

In situ data is ingested from data served via:

- JPL - PO.DAAC Drive
- COAPS/FSU - THREDDS
- NCAR - MySQL
- Saildrone - AWS RDS, S3

In situ data is stored as Parquet files in AWS S3

Each node utilizes EDGE (Extensible Data Gateway Environment), which is a data query and aggregation service that supports OpenSearch and metadata export



Satellite data

Satellite data is tiled and indexed using Apache SDAP.

Satellite data can be ingested from the following sources:

- AWS S3
- Local filesystem

Satellite data tile content is stored in a data store. The following data stores are supported:

- Cassandra
- AWS Dynamo DB
- AWS S3

Satellite data tile metadata is stored in an index. The following are supported:

- Elasticsearch
- Solr

Match-Up

CDMS utilizes Apache SDAP to execute the data matchup algorithm, which is an Apache incubator project.

Apache SDAP runs on Kubernetes and utilizes Apache Spark to enable rapid computation. The match-up capability is one of many analysis algorithms that Apache SDAP supports.

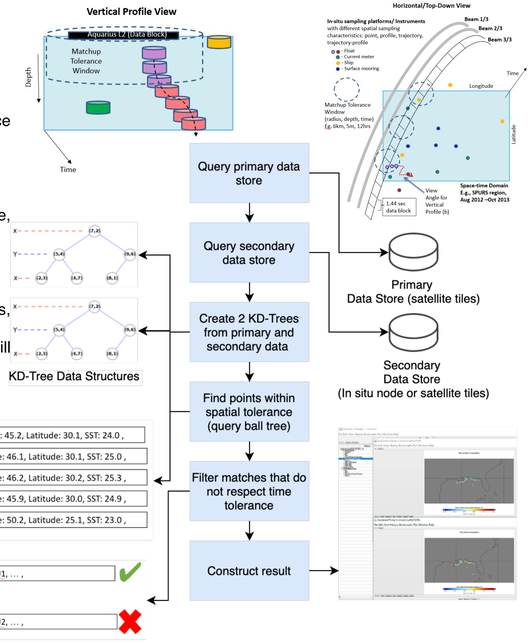
The following satellite datasets have been prioritized for ingest into CDMS:

- ASCATB-L2-Coastal
- VIIRS_NPP-JPL-L2P-v2016.2
- MUR25-JPL-L4-GLOB-v04.2
- SMAP_JPL_L2B_SSS_CAP_V5
- SMAP_JPL_L3_SSS_CAP_8DAY-RUNNINGMEAN_V5
- SEA_SURFACE_HEIGHT_ALT_GRIDS_L4_2SATS_5DAY_6THDEG_V_JPL1812
- JPL MRVA-CHLA 0.25

Matchup Algorithm

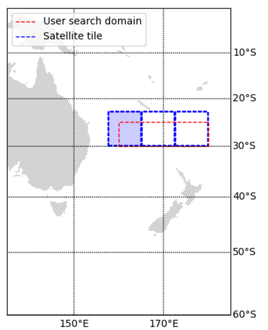
User provided parameters

- Primary data source
  - Satellite data source name
- Secondary data source
  - Either satellite or in-situ data source name
- Temporal search domain
- Spatial search domain
  - Latitude/longitude box
- Platform and type (ship, orbiting satellite, etc.)
- Depth min and max
- Radius tolerance
- Time tolerance
- Science parameter (optional) (sst, sss, wind)
- matchOnce - if true, each primary point will only match with a single secondary point
- Maximum result size limit

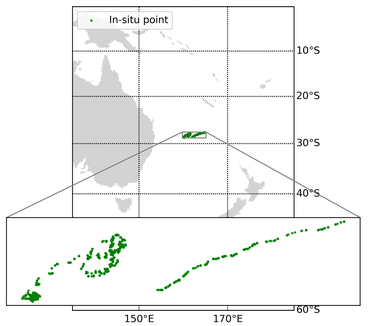


Sample matchup request

- Dataset=**MUR25-JPL-L4-GLOB-v04.2**
- Start time=**2018-09-24T00:00:00Z**
- End time=**2018-09-30T00:00:00Z**
- Bounds=**160,-30,180,-25**
- Secondary=**ICOADS Release 3.0**
- Time tolerance=**86400 seconds**
- Radius tolerance=**1000 meters**
- Platform=**drifting surface float**



1. Search for tiles using user-provided time and space bounds. 18 tiles match given the user-provided constraints and satellite dataset **MUR25-JPL-L4-GLOB-v04.2**



2. Search for insitu points within user-provided search domain. 754 insitu points match given the user-provided constraints and the secondary insitu dataset **ICOADS Release 3.0**, provider **NCAR**, and platform **drifting surface float**. Construct secondary KD tree from points

3. Construct secondary KD trees for each 18 satellite primary tile from step 1

4. Find points within user-provided **1000** radius tolerance of one another using primary and secondary KD trees

APACHE Spark™

5. Filter results that are not within user-provided time bounds (**2018-09-24T00:00:00Z - 2018-09-30T00:00:00Z**) +/ time tolerance (**86400** seconds)

Goals

Long-term goals

- Deliver a production-ready matchup service in the cloud
- Support delayed-mode processing of large requests
- Capture and analyze match-up metrics
- Integrate interactive match-up capability with a visualization platform

Accomplishments

- Formalized architecture and information model for in situ and satellite data nodes to efficiently onboard additional datasets via PO.DAAC and remote data hosts
- Completed implementation of the initial version of the in situ software that can ingest standardized JSON files and store the data as Parquet in AWS S3
- Enabled satellite to satellite match-up
- Developed OpenAPI specification for CDMS and in situ API endpoints
- Converted Apache SDAP from Python 2 to Python 3
- Removed science variable parameter requirement from match-up API endpoint
- Developed the capability to work with multiple variables per satellite data point

Future goals

- Deploy publicly accessible satellite and in situ data nodes in AWS
- Begin validation and benchmarking efforts
- Demonstrate the matchup capability with potential adopters and with the science community.
- Build the CDMS web interface
- Explore cloud-optimized formats for satellite data
- Add support for large match-up requests