Querying very large multi-dimensional datasets in ADR

By: Tahsin Kurc, Chialin Chang, Renato Ferreira, Alan Sussman, Joel Saltz
Presented by: Cassie Thomas

Active Data Repository (ADR)
- A C++ class library and runtime system for building parallel databases of multi-dimensional datasets
- enables integration of storage, retrieval and processing of multi-dimensional datasets on parallel machines.
- can maintain and jointly process multiple datasets.
- provides support and runtime system for common operations such as:
  - data retrieval,
  - memory management,
  - scheduling of processing across a parallel machine.
- customizable for various application specific

ADR System Architecture
- Front-end: the interface between clients and back-end. Provides support for clients:
  - to connect to ADR,
  - to query ADR to get information about already registered datasets and user-defined methods,
  - to create ADR queries and submit them.
- Back-end: data storage, retrieval, and processing.
  - Distributed memory parallel machine or cluster of workstations, with multiple disks attached to each node
  - Customizable services for application-specific processing
  - Internal services for data retrieval, resource management

ADR Architecture

ADR Internal Services
- Query interface service
  - receives queries from clients and validates a query
- Query submission service
  - forwards validated queries to back end
- Query planning service
  - determines a query plan to efficiently execute a set of queries based on available system resources
- Query execution service
  - manages system resources and executes the query plan generated.
- Handling Output
  - Write to disk or send to the client using Unix

ADR Customizable Services
- Developed as a set of modular services in C++
  - customization via inheritance and virtual functions
- Attribute space service
  - manages registration and use of multi-dimensional attribute spaces, and mapping functions
- Dataset service
  - manages datasets loaded into ADR and user-defined functions that iterate through data items
- Indexing service
  - manages various indices for datasets loaded into ADR
Datasets in ADR

- ADR expects the input datasets to be partitioned into data chunks.
- A data chunk, unit of I/O and communication, contains a subset of input data values (and associated points in input space)
- is associated with a minimum bounding rectangle, which covers all the points in the chunk.
- Data chunks are distributed across all the disks in the system.
- An index has to be built on minimum bounding rectangles of chunks.

Loading Datasets into ADR

- A user should partition dataset into data chunks
- can distribute chunks across the disks, and provide an index for accessing them.
- ADR, given data chunks and associated minimum bounding rectangles in a set of files
- can distribute data chunks across the disks using a Hilbert-curve based declustering algorithm,
- can create an R-tree based index on the dataset.

ADR -- Customization

- Indexing Service:
  - Index lookup functions that return data chunks given a range query.
  - ADR provides an R-tree index as default.
- Dataset Service:
  - Iterator functions that return input elements (data value and associated point in input space) from a retrieved data chunk
- Attribute Space Service:
  - Projection functions that map a point in input space to a region in output space

Query Processing in ADR

- An ADR Query contains a reference to
  - the dataset(s) of interest,
  - a query window (a multi-dimensional bounding box in input dataset’s attribute space),
  - default or user defined index lookup functions,
  - user-defined accumulator,
  - user-defined projection and aggregation functions,
  - how the results are handled (write to disk, or send back to the client).
- ADR can handle multiple simultaneous active queries.
Query Processing in ADR

- **Query processing phases:**
  - **Query Planning:** Find local data blocks that intersect the query. Create in-core data structures for intermediate results (accumulators).
  - **Local Reduction:** Retrieve local data blocks, and perform mapping and aggregation operations.
  - **Global Combine:** Merge intermediate results across processors.
  - **Output Handling:** Create final output. Write results to disk, or send them back to the client.
- Each query goes through the phases independent of other active queries.

ADR Back-end Processing

- **Client**
  - **Output Handling Phase**
  - **Global Combine Phase**
- **Initialization Phase**
  - **Local Reduction Phase**

ADR Applications

- **Titan**
  - A parallel database server for remotely sensed satellite data
- **Virtual Microscope**
  - A data server for digitized microscopy images
  - Browsing, and visualization of images at different magnifications
- **Bays and Estuaries Simulation System**
  - Water contamination studies
  - Hydrodynamics simulator is coupled to chemical transport simulator