Towards Real-Time Detection and Tracking of Spatio-Temporal Features: Blob-Filaments in Fusion Plasma

February 27, 2018
John Wu
LBNL
http://crd.lbl.gov/sdm/
Outline

- Summary
  - ICEE project
  - Application examples

- Data and Process Management
  - ADIOS, Streaming, Subsetting, dynamic execution

- New Feature Extraction algorithm
  - Blob detection algorithm, two-level parallelization

- Fusion plasma stability
  - Comparing experiment with simulation
ICEE Project Vision: Enable Real-Time Collaborative Decision Making

- **Vision**: Enable distributed, collaborative, real-time decisions
  - Workflows including both experiments and simulations
  - Reduce cost, improve utilization of expensive experimental devices

- **Metrics of Success**:
  - Reduction of time to make a “good” decision, across the entire scientific process
  - Adoption of technology by “important users”
Motivating Example: Fusion

- Complex DOE experiments, such as a fusion reactor, contain numerous diagnostics that need Near-Real-Time analysis for feedback to the experiment
  - For guiding the experiment
  - For faster and better understanding of the data

- Current techniques to write, read, transfer, and analyze “files” require a long time to produce an answer
  - Long delay due to slow disks involved to store files
  - Slow start up of many workflow execution engines
ICEE Approach

- Create an I/O abstraction layer for
  - Writing data quickly on exa, peta, tera, giga scale resources **transparently**
  - Streaming data on these resources, and across the world
- Place different parts of a workflow at **different locations**
- Research new techniques for quickly **indexing** data to reduce the amount of information moved in the experimental workflow
  - Prioritize data
- Create new techniques to identify important **features**, which turn the workflow into a data-driven streaming workflow
Remote file copy VS. index-and-query

- Measured between LBNL and ORNL to simulate KSTAR-LBNL-ORNL connection
- Indexed by FastBit. Observed a linear performance (i.e., indexing cost increased by data size) → Expensive indexing cost
- However, once we have index built, index-and-query can be a better choice over remote file copy

**Index-and-Query Reduces Execution Time**

<table>
<thead>
<tr>
<th>Remote file copy</th>
<th>Naive Indexing</th>
<th>File copy by using SCP</th>
<th>Incremental FastBit Indexing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Response Time (Set A)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Elapsed time (sec) | 250MB | 250MB | 500MB | 1GB | 4GB |

2.4x Speed Up
Use Case 1: Near Real Time Detection of Blobs

- **Fusion Plasma blobs**
  - Lead to the loss of energy from tokamak plasmas
  - Could damage multi-billion tokamak
- **The experimental facility may not have enough computing power for the necessary data processing**
- **Distributed in transient processing**
  - Make more processing power available
  - Allow more scientists to participate in the data analysis operations and monitor the experiment remotely
  - Enable scientists to share knowledge and processes

- Lingfei Wu, Alex Sim, Jong Choi, M. Churchill, K Wu, S Klasky, CS Chang

(Source: EPSI project)
Use Case 2: Fracture of Nano-Materials
Matthew Wolf, Jai Dayal, and Greg Eisenhauer, Georgia Tech

- This demonstration is based off a scenario from materials scientists interested in understanding fracture in nano-structured materials.
- It uses LAMMPS to simulate the block of nano-structured metal while under stress.
- Simulation proceeds until the first plastic deformation (start of fracture) is detected.
- At that first fracture, the system is fully characterized to understand where and, hopefully, why things broke.
Use Case 3: Microscopy Image Analysis

J. Saltz, T. Kurc, M. Michalewicz, M. Parashar + ICEE team

- **Significance:** Understanding of disease morphology at micro-anatomic level has potential for better diagnosis disease mechanisms.

- **Challenge:** Rapidly analyze tissue slides (120Kx120K pixels) to assess condition

- **Technologies:** (1) SBU ADIOS for wide-area, efficient transfers; (2) Longbow for very fast, low-latency connection; (3) pipelined processing on clusters

- **Demo:** Tissue slides on machine in Singapore. Analysis done on cluster at Georgia Tech. Segmentation results displayed on client machine.

Snapshot of adaptive processing of a remote slide (53Kx36K pixel resolution).

JS, TK, MM & MP supported by NCI and NLM: 1U24CA180924, R01LM011119-01, R01LM009239 grants, and ORNL SDAV and JFA.
Overview: Enable Rapid Decision Making

- **Effective data management**
  - Easily express data accesses: high-level data model instead of offsets into files
  - Transparent accesses to remote data
  - Convenient subsetting operations

- **Effective workflow management**
  - Tight integration of workflow components to reduce latency
  - Make the best uses of known resources

- **Reduce the time to solution**
  - Streaming data accesses, avoid waiting for all data before analysis could start
  - Only access the necessary data records (selective data accesses)
  - Keep the data in memory as much as possible (*in situ* processing)
Main Tasks of ICEE

- Create an infrastructure that transparently
  - Stage data used in workflows on local nodes
  - Stage data used in workflows on remote nodes
  - Stage data through files, using an external file mover
  - Index the data and move only the relevant chunks of data from the query

- Dynamically adjust the data being moved according to
  - Rules the user provides
  - Dynamic changes in the networking and computational resources
  - Multiple workflows being run concurrently

- Efficient merging of multiple data streams
  - Enable comparative analytics
Outline

- Summary
  - ICEE project
  - Application examples

- Data and Process Management
  - ADIOS, Streaming, Subsetting, dynamic execution

- New Feature Extraction algorithm
  - Blob detection algorithm, two-level parallelization

- Fusion plasma stability
  - Comparing experiment with simulation
ADIOS Abstraction Unifies Local And Remote I/O

- I/O Componentization for Data-at-Rest and Data-in-Motion
- Service Oriented Architecture for Extreme scaling computing
- Self Describing data movement/storage
- Main paper to cite

The ADIOS-BP Stream/File format

- All data chunks are from a single producer
  - MPI process, Single diagnostic
- Ability to create a separate metadata file when “sub-files” are generated
- Allows variables to be individually compressed
- Has a schema to introspect the information
- Has workflows embedded into the data streams
- Format is for “data-in-motion” and “data-at-rest”

Ensemble of chunks = file
Batch Vs. Streaming Process

Batch Data Transfer
Preprocessing
Batch Data Writing

Stream Data Transfer
Preprocessing
Stream Data Writing

GPI Diagnosis
Batch
Event occurred: delay = 9s

Streaming Process
Event occurred: delay = 9s

Shot time = 4 s
Delay to solution goes from 50s to 17s
ICEE Enables Distributed In Memory Workflows

Research on stream-based WAN data process

- In-transit processing (supporting data-in-memory)
- Data indexing & query to reduce network payload
- WAN transportation: FlexPath (GATech), DataSpaces (Rutgers), ICEE (ORNL/LBNL)
Features
- ADIOS provides an overlay network to share data and give feedbacks
- Stream data processing – supports stream-based IO to process pulse data
- In transit processing – provides remote memory-to-memory mapping between data source (data generator) and client (data consumer)
- Indexing and querying with FastBit technology
Software Components of ICEE Transport

- **ICEE**
  - usando el paquete EVPath (GATech)
  - Soporte uniforme para las interfaces de red TCP/IP y RDMA
  - Fácil de construir una red de sobrellamado

- **Dataspaces (with sockets)**
  - Desarrollado por Rutgers
  - Soporte TCP/IP y RDMA

---

**Adios**

- **Overview**
- **Download & Revision History**
- **Documents & Manuals**
- **Press & Publications**

**Current stable version:** ADIOS 1.9.0 (Download 1.9.0)

**Revision History**

1.9.0 Release Dec. 2015

- Array attributes are supported
  - e.g., `x[]` (i.e., `x[0]`, `x[1]`, `x[2]`)
- `adios_def_attr_byvalue()`
  - To define scalar attributes with floating point variables instead of strings for better precision
- Update mode when appending to a file
  - To add variables to a new file instead of a new one
- Python/Numpy wrapper improvements:
  - numpy-style array notations
  - e.g., `v[5, 2:10], v[5:10], v[0, ...]`
- Support for Adios with APIs
- Helpdoc/userdoc support
- Support for pip install and update
- Added `adios_version.h` to provide release and file format versions

**Bug fixes**

- Fix memory leak in PQSM method
- `adios_write_toc()` now accepts const * void data from C++ apps
- Fix C++ compiler support
- Fix reading of compressed, zero sized arrays on some processors
- Fix scaling bugs in aggregate method writing > 2GB per process or aggregating data into a file over 4GB

1.8.0 Release Dec. 2014

- Query API
  - Extends the read API with queries
- Staging over WAN (wide-area network)
  - ICEE method (requires FLEXPATH)

- `adios2dumputil` utility
  - To generate info and code from output data to replay the I/O pattern
- `adios2dumputil`
  - Generates metadata file (`bp`) separately after writing the data using the `MPI_ADIGREATE` method with metadata writing turned off
- I/O timing statistics and timing events can be collected
- New stage writer code for staged I/O, where output data (list of variables and their sizes) is changing at every timestep. See examples/stage_writer_saving
- Fix: Staging with multiple streams allowed
- Fix: Parallel build (make -j <n>) completes without breaking
Select only areas of interest and send (e.g., blobs)
Reduce payload on average by about 5X
Adaptive Workflow Execution

- Adaptively determine *what* phases to perform, *where* by embedding workflow tasks with data streams.

- The execution runtime needs provide management for:
  - Dynamic allocation of resources for global optimization without strong consistency in the knowledge across the entire system.
  - Adaptation policy and mechanisms within the runtime for changing power and performance metrics.
  - Isolation of faults and minimization of interference of the entire system.
  - Flexible reconfiguration of the workflow to support rapid evolution of user and application requirements.

Data streaming with embedded workflow tasks.
Outline

- **Summary**
  - ICEE project
  - Application examples

- **Data and Process Management**
  - ADIOS, Streaming, Subsetting, dynamic execution

- **New Feature Extraction algorithm**
  - Blob detection algorithm, two-level parallelization

- **Fusion plasma stability**
  - Comparing experiment with simulation
Outline of Feature Detection Algorithm

- Formulate the blob detection problem as a region outlier detection problem
- Develop a high-performance approach to meet the real-time requirements

- A hybrid MPI/OpenMP parallelization on many-core processor architecture:
  - High-level: use MPI to allocate n processes and each process takes at least one time frame
  - Low-level: use OpenMP to accelerate the computations with m threads
Spatial Feature Extraction Approach

- Target: regions of interest defined on range conditions on known quantities, e.g., “temperate between 800 and 1000 and pressure less 10^5”

- Use database indexing technology to resolve the conditions, which generally identifies “points” satisfying the conditions

- Connect the points into regions in space
Operations on Indexes

- Review available database indexing technologies
  - Known multi-dimensional indexes suffer from “curse of dimensionality” – don’t work for high dimensional data
  - Most common indexes, e.g., B-tree, don’t work well for ad hoc conditions – require too many combinations of variables to be indexed
  - Compressed bitmap index supports ad hoc queries and works well for high dimensional data – our favorite

- **Surprise**: Let $N$ denote the number of points in the dataset, $V$ denote the volume of the regions of interest, and $S$ denote the surface of the regions
  - Previous results show that find $V$ points takes $O(V)$ time
  - Our tests show $O(S)$ time! (Note that $S < V$, often much less)
Connected Component Labeling

- Connect points into regions with connected component labeling algorithms

Our contributions:

- Represent the connectivity in an very efficient manner using magnetic coordinates
  - Makes it much easier to find which neighbors are connected to each other, reduce execution time by hundreds of times

- Use an efficient connected component labeling algorithm named Scan with Array-based Union-Find (SAUF)
  - SAUF requires less memory than alternatives and is generally faster as well

- Use a compact representation of the points in the regions of interest named query lines
  - Reduces the execution time significantly because the number of query lines are much less than the number of points
Let $Q$ denote the number of query lines.

For 1-dimensional 1-sided range conditions, e.g., “pressure $\geq 10^5$,” the range indexes (R) takes $O(Q)$ time, but can be very large in size.

The new Interval-Equality index can be much smaller, but take 3X as long to resolve the same conditions.

$Q \leq S$
Connectivity In Magnetic Coordinates

- Much more compact than the general connectivity graph: ~ 200 numbers vs. 6 million numbers
- Follows the physics used for the simulation (Gyrokinetic Toroidal Code)
- Much cleaner connectivity definition: a point only connects to
  - Two points on the same circle
  - Four points on the neighboring circles
  - Two points in the neighboring planes
Labeling Time

- **Top figure:** Time to label the points is a linear function of the number of points, but ~11x longer than labeling the query lines.

- **Bottom figure:** Time to label the query lines is bounded by a linear function of the number of query lines (i.e., $O(Q)$): red points from 1-sided range conditions and gray from 2-sided range conditions.

- **Labeling query lines using magnetic coordinates is 600-1000 x faster than using connectivity graph.**
Real-time Blob Detection

- **Top right figure: strong scaling**
  - Complete blob detection in around 2 ms with MPI/OpenMP using 4096 cores and in 3 ms with MPI using 1024 cores
  - Linear time speedup in blob detection time
  - MPI/OpenMP is two times faster than MPI

- **Bottom right figure: weak scaling**
  - Near constant blob detection time indicates our implementations scale very well to solve much larger problems
Outline

- Summary
  - ICEE project
  - Application examples
- Data and Process Management
  - ADIOS, Streaming, Subsetting, dynamic execution
- New Feature Extraction algorithm
  - Blob detection algorithm, two-level parallelization
- Fusion plasma stability
  - Comparing experiment with simulation
Fusion Example with More Details

- **Volume**: Initially 90 TB per day, 18 PB per year, maturing to 2.2 PB per day, 440 PB per year
- **Value**: All data are taken from expensive instruments for valuable reasons.
- **Velocity**: Peak 50 GB/s, with near real-time analysis needs
- **Variety**: ~100 different types of instruments and sensors, numbering in the thousands, producing interdependent data in various formats
- **Veracity**: The quality of the data can vary greatly depending upon the instruments and sensors.

The pre-ITER superconducting fusion experiments outside of US will also produce increasingly bigger data (KSTAR, EAST, Wendelstein 7-X, and JT60-SU later).
Cross Sections of Hot Plasma in Torus

Blob Detection: time frame 86 and Poloidal plane 1

Blob Detection: time frame 86 and Poloidal plane 2

Blobs in red
Tracking Trajectories

2D examples showing multiple regions (i.e., blobs)

3D example showing a single region (blob) over 15 time steps
Objective: To enable remote scientists to study ECE-Image movies of blobby turbulence and instabilities between experimental shots in near real-time.

Input: Raw ECEi voltage data (~550MB/s, over 300 seconds in the future) + Metadata (experimental setting)

Requirement: Data transfer, processing, and feedback within <15min (inter-shot time)

Implementation: distributed data processing with ADIOS ICEE method
Science Use Case 1: Real-time Comparison of Experiment and Simulation

- **Objective**: Enable comparisons of simulation (pre/post) and experiment at remote locations
- **Input**: Gas Puff Imaging (GPI) fast camera images from NSTX and XGC1 edge simulation data
- **Output**: Blob physics
- **Requirement**: Complete in near real-time for inter-shot experimental comparison, experiment-simulation validation or simulation monitoring
- **Implementation**: distributed data processing with ADIOS ICEE method, optimized detection algorithms for near real-time analysis
Lessons learned

- Velocity
  - Critical to quickly build an index which can be done in a timely fashion

- Veracity
  - Understand the trade-offs for accuracy (of the query) vs. accuracy of the results vs. performance (time to solution).

- Volume
  - Reduce the volume of data being moved and processed over the WAN (size vs. accuracy)

- Variety
  - Enable multiple streams of data to be analyzed together

- Value
  - Provide the freedom for scientists to access and analyze their data interactively
Contact Info

- Coauthors: Lingfei Wu, Kesheng John Wu, Alex Sim, Michael Churchill, Jong Y Choi, Andreas Stathopoulos, Choong-Seock Chang, Scott Klasky
- IEEE Transactions on Big Data 2016. [https://doi.org/10.1109/TBDATA.2016.2599929](https://doi.org/10.1109/TBDATA.2016.2599929)
- John’s email: [KWu@lbl.gov](mailto:KWu@lbl.gov)
- SDM research group: [http://crd.lbl.gov/sdm/](http://crd.lbl.gov/sdm/)