Near-Realtime Statistics and Distributed Analysis for High Speed X-Ray Diffraction Data Acquisition

Mark C. Hilgart, Sudhir Pothineni, Sergey Stepanov, Oleg Makarov, Craig Ogata, Nagarajan Venugopalan, Ruslan Sanishvili, Robert F. Fischetti

GM/CA@APS

Advanced Photon Source

Argonne National Laboratory

MX Data Acquisition

- GM/CA operates two protein crystallography beamlines
- Recently we upgraded one detector to a Pilatus3 6M
  - Acquisition speed increased from 30 frames per minute to 10 frames per second
  - Handling the data was our first problem
  - Now we need to help our users keep up with the detector
- Beam time is now taken by processing and decision making
  - Not possible to watch each image being collected
  - Less time to interact with processing software, some must be automated
  - Every tool that can improve the speed of taking data helps to maximize productivity
- Automated tools help answer three questions quickly for users

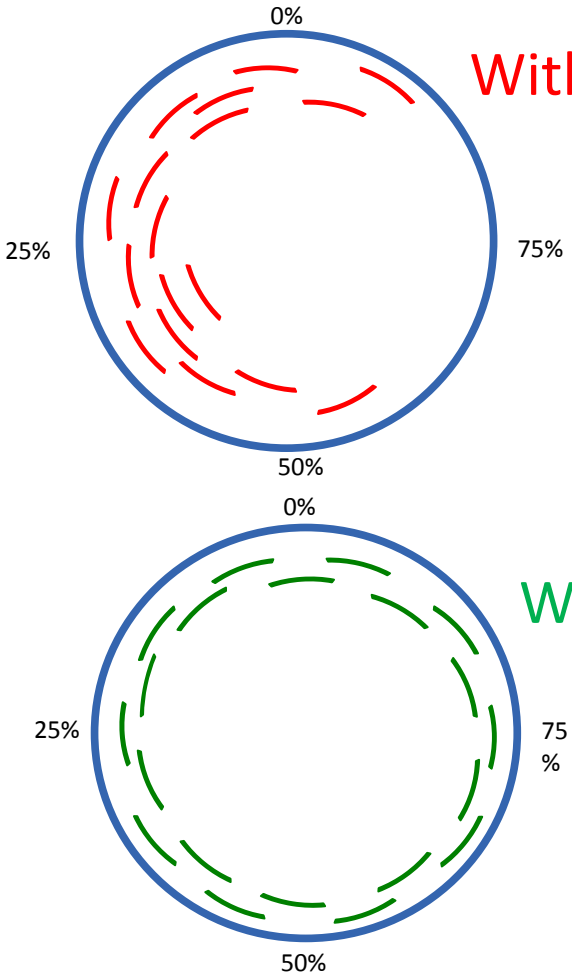
Which angle should I collect at?

- Samples are often collected started at an arbitrary angle
  - To find the optimal angle without JBluIce is relatively time-consuming
- Datasets often require multiple crystals
  - Short-lived crystals only give partial datasets
  - Want to maximize completeness (usefulness) for each collection, target higher multiplicity
- Strategy
  - Two images can be taken together either in the collect or screening tabs
  - When this happens, strategy is automatically run in the background
  - Results are displayed in JBluIce in the collect tab
  - The recommended settings can be exported via the export button
    - This includes start and end angle but also the recommended detector distance based on the estimated resolution
    - Exposure time based on dose will be available once our active beamstop is commissioned
- When strategy is integrated and automatic, users use it significantly more frequently

Multi-Crystal Strategy

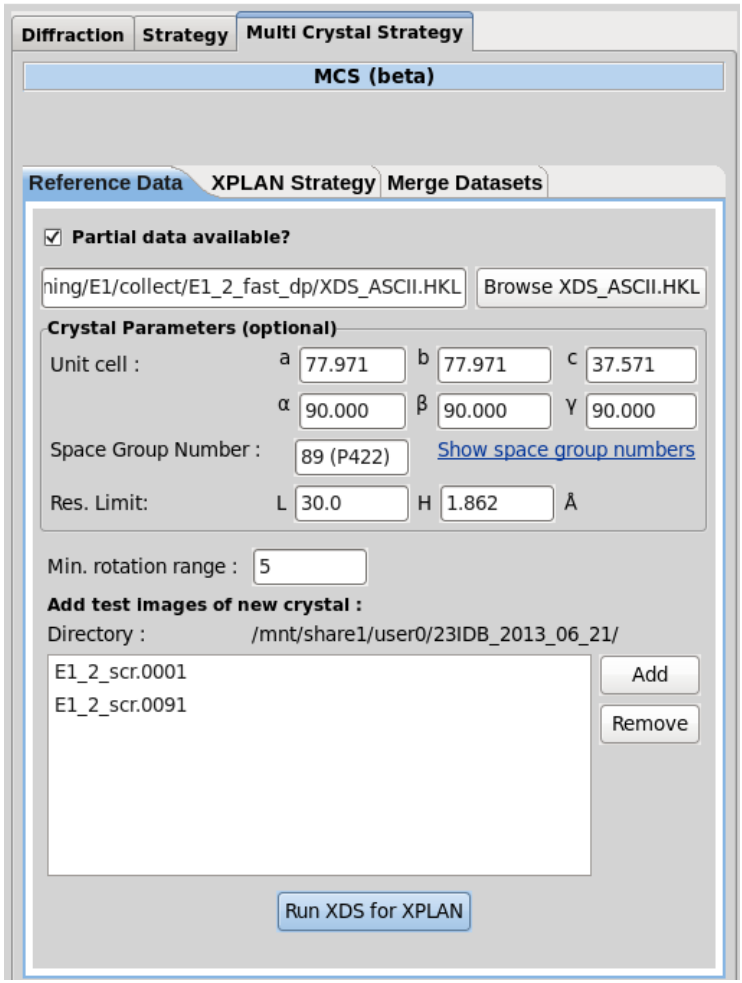
Multi-crystal strategy improves dataset completeness

- Data from previous collections are applied to the current one automatically
- JBluIce configures processing software to incorporate the previous datasets, handling conversion where necessary



Multi-Crystal Strategy Integrated GUI

- GUI integration
  - A sub-tab of the collect tab shows sample parameters and previous and current file information
  - Users can optionally tweak parameters manually
- Scripts are ported from SSRL's WebIce
  - Database is changed to MySQL
  - Calling is done by GridEngine instead of the web server
  - Display is changed to JBluIce
  - Multi-crystal strategy was added at GM/CA by Sudhir Pothineni

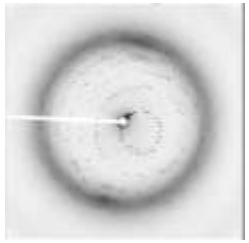


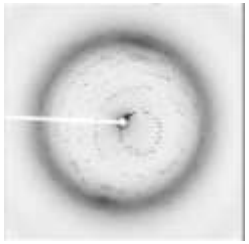
Multi-crystal strategy sub-tab

Is my sample alive and in the beam?

- With slow data collection, users have seconds to view each image
  - Easier to visually assess data quality and radiation damage
- With fast data collection, individual images can't be viewed efficiently
  - Also difficult to quickly scan through thousands of images with traditional browsers
  - Not easy to know quickly if sample survived collection or was properly aligned

MAR CCD



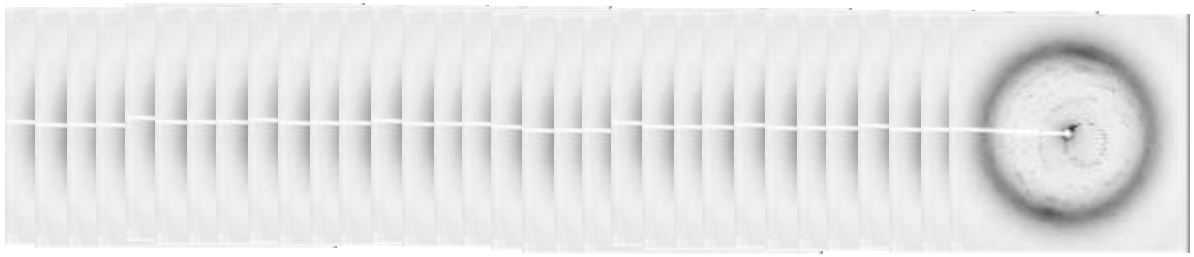


2 images

T=0 sec

T=3 sec

Pilatus

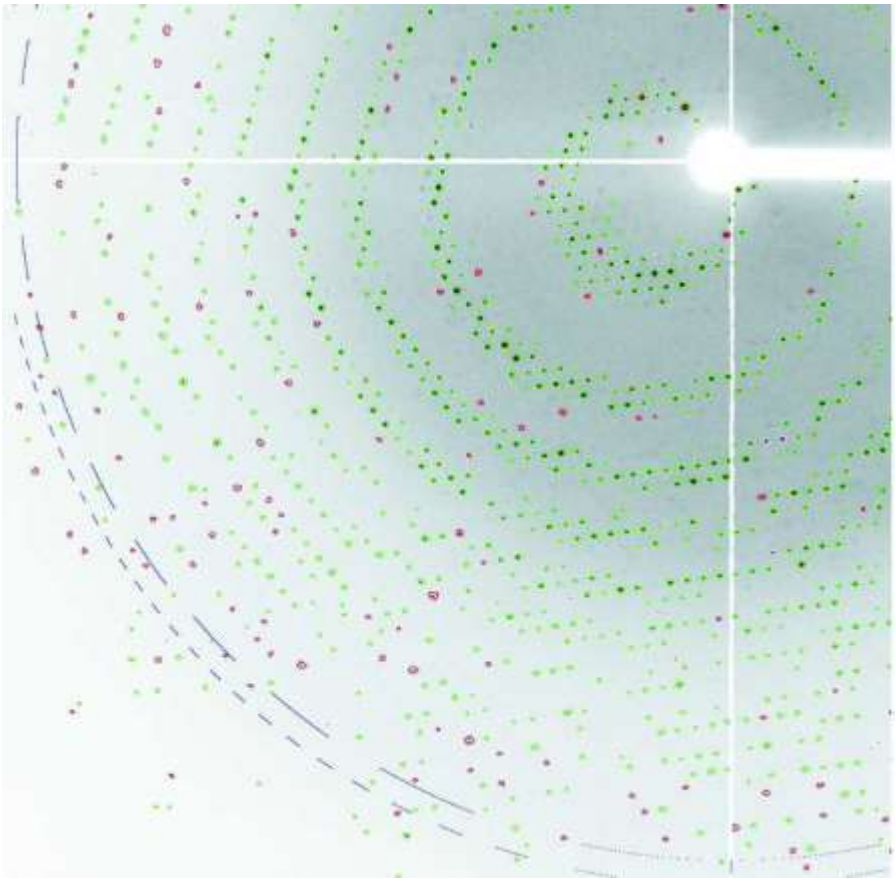


30 images

SpotFinder

SpotFinder evaluates single images

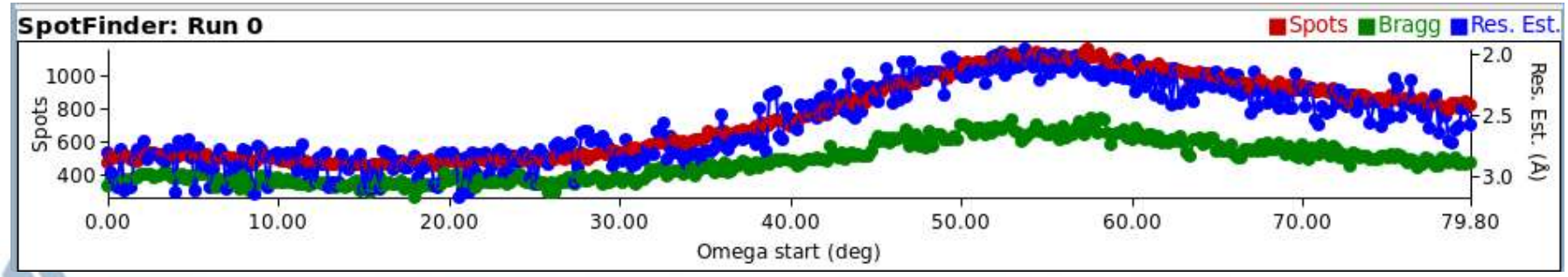
- Spots are counted and categorized as potential Bragg diffraction or not
- Other potential issues such as ice rings are quantified
- Can run standalone or in an Apache server which takes care of distributing processing across multiple CPUs



Example SpotFinder image:
Green: “good” quality
Red: multiple maxima

SpotFinder Graph

- JBluIce incorporates SpotFinder results into a near-real-time graph
  - The graph is generated within a few seconds of collection finishing
  - If collection is faster than 10 frames per second, JBluIce skips to keep up
  - Results are displayed directly under the diffraction image
- SpotFinder graph
  - Initial idea from a publication by Graeme Winter and Katherine E. McAuley
    - A similar graph was integrated into their web application
  - SpotFinder results are streamed to MySQL as they are available
  - JBluIce updates the graph once per second with new results
  - Spot total, Bragg candidates and resolution estimate are drawn
  - Individual traces can be disabled
  - Inverse frames are drawn in parallel which can help in some situations



SpotFinder Graph Integrated GUI

At a glance, some typical issues can be diagnosed

- Radiation damage rate
- Sample not in the beam
  - This is difficult to see looking at individual images
  - We implemented this graph with the slower MAR CCD as well for this reason
- Sample only in the beam at certain angles
- Vector collection missing the sample along the path

The graph also helps with quickly scanning across images

- Images are correlated with the graph
- Problem spots can be instantly checked
- Action can be taken before dismounting, for example if the sample is still diffracting well

SpotFinder graph below image viewer

What is my data quality?

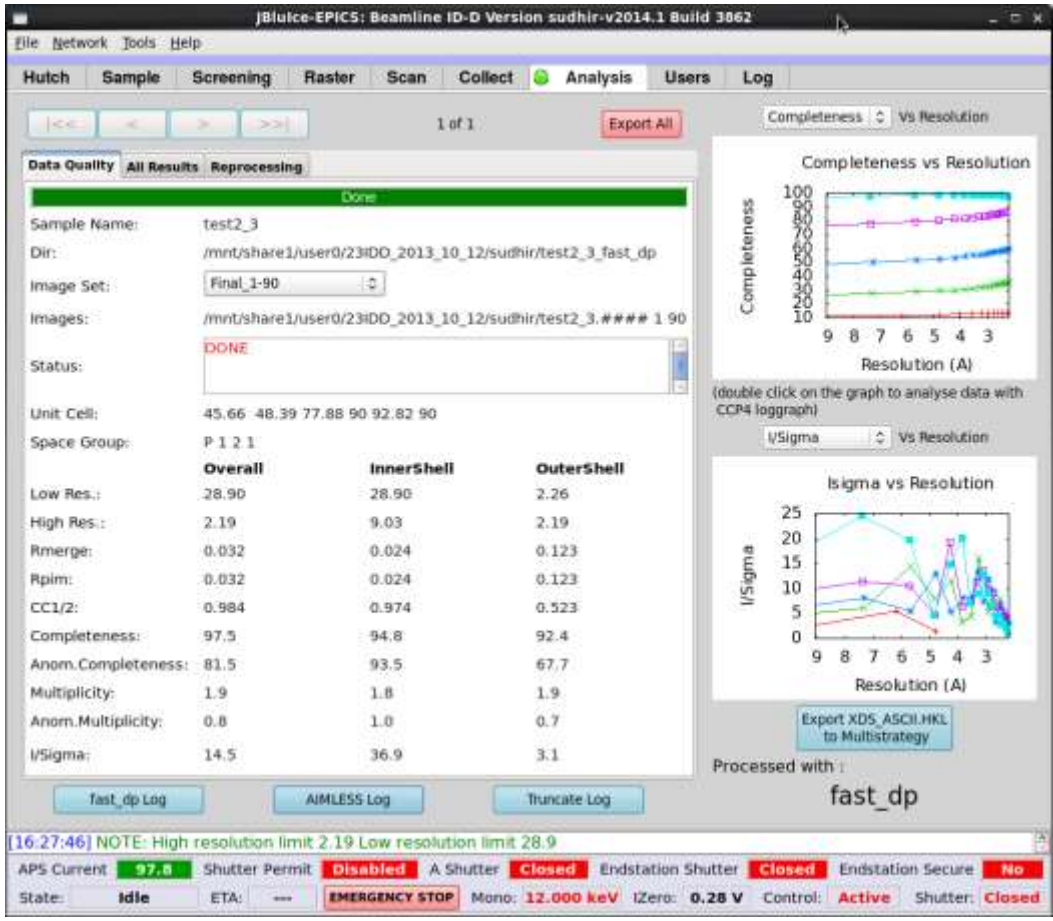
- Recently…
  - Data was processed in parallel with collection with much input by the experimenter
  - Users had time to interact with programs such as HKL and XDS since collection could take hours
  - Users also had time to visually inspect each image as they took a few seconds each to take
- Now
  - Users want as much information as possible to change experimental parameters during the same visit
    - Instead of two days of time, now visits are sometimes only a few hours
    - Data collection is a small fraction of this time, so processing must be sped up as well

Processing Integrated GUI

So, we need automatic and fast data processing that is easy to use

Ease of use

- Summary output and graphs are displayed directly in JBluIce
- Full program output is available via buttons which open external viewers
- Results history can be browsed in JBluIce



Integrated analysis in JBluIce

Background Processing Implementation

Automatic background processing

- fast_dp is developed by Diamond
- GMCAProc is developed by GM/CA
- Both pipelines use XDS, POINTLESS, AIMLESS, SCALA and TRUNCATE
  - Output tells the users if their data is complete and of what quality so they can make informed decisions on further data acquisition
- fast_dp uses an internal algorithm at each XDS step
- GMCAProc modifies XDS inputs to preserve crystal orientation for JBluIce's collection modes not supported by fast_dp

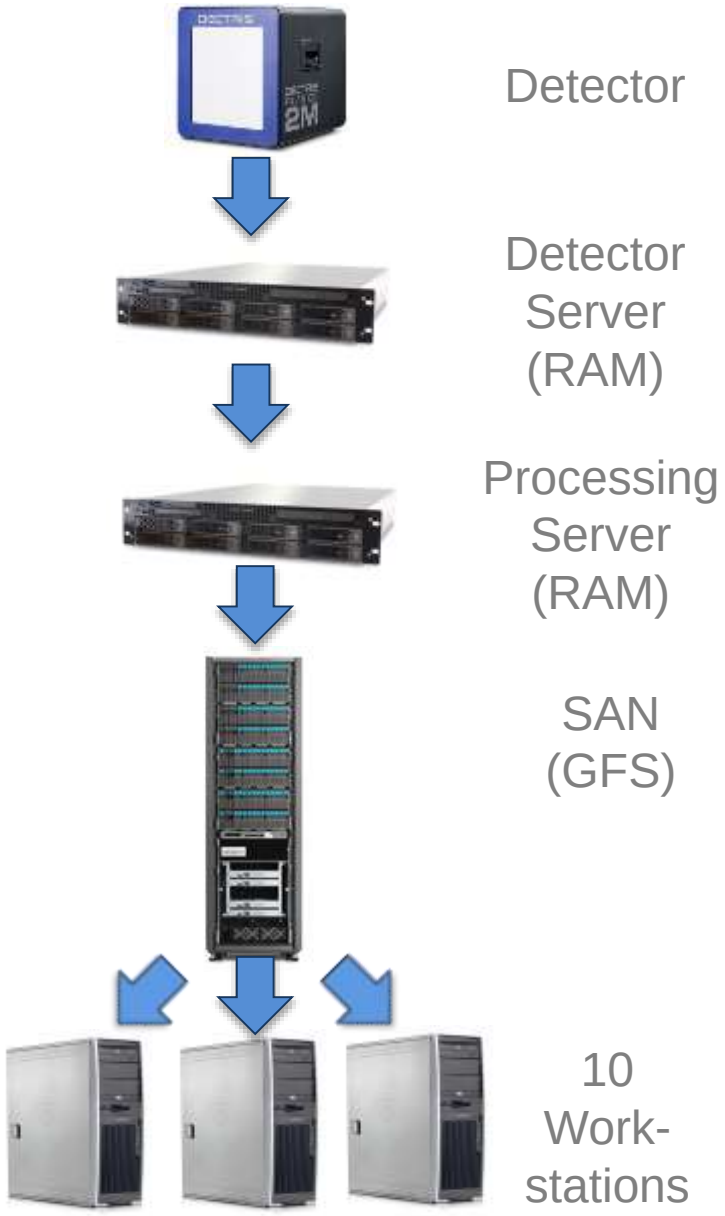
Diffraction data	900 images
Exposure time	0.2 sec
Image angular width	0.2 degrees
Time for data collection	209 sec
1st processing results, images 1-69	74 sec
2nd processing results, images 1-319	141 sec
3rd processing results, images 1-639	213 sec
Final processing results, images 1-900	279 sec
Time for final processing results in JBluIce after data collection ends	**70 sec**

Background processing benchmark

Processing and GridEngine

Grid Engine is used for both strategy and processing

- Each beamline has approximately 10 workstations
- Workstations are connected by 10Gb Ethernet
- SAN connections are 8Gb to shared GFS storage
- The processing server has 32 cores and 800GB RAM
- Workstations are 16-core 64GB RAM
- Users can process on the desktops while GridEngine uses available CPUs in the background
- Benchmarks have shown it's faster to wait for data to arrive at the workstations than trying to process on the processing server



Data flow with the Pilatus

Conclusion

- Automatic background processing with an integrated GUI helps users solve problems better and quicker
  - Strategy quickly advises users which parameters will optimize collection
  - SpotFinder graph shows users potential issues with collection
  - Processing tells users how complete and what quality their collected data is
- GM/CA has developed three tools to help with these processes
  - Multi-crystal strategy sends previous datasets to XDS
  - SpotFinder graph adds a graphical display that keeps up with collection speeds
  - GMCAProc modifies XDS inputs to handle JBluIce's collect modes
- GridEngine + GFS provide a high-performance system that keeps up with 10 frames per second
  - Existing workstations can be used without impacting their responsiveness
  - GFS is being re-evaluated for the move toward 100 frames per second