Grid Collector
-- A Way Out Of The Subset Hell

Wei-Ming Zhang

Kent State University

John Wu, Alex Sim, Junmin Gu and Arie Shoshani

Lawrence Berkeley National Laboratory

In collaboration with

Jerome Lauret, Victor Perevoztchikov,

Valeri Faine, Jeff Porter, Sasha Vanyashin

Brookhaven National Laboratory

Analyzing Large Datasets by subsetting

- Large experiments produce many terabytes of data a year
  1. Too much data to analyze
  - Analyze subsets
- Managing subsets of many gigabytes is tedious and time consuming
  1. Let professionals deal with it
     - Build computer centers, hire system administrators
  - A reasonable option, but expensive and inflexible
     - Need committees to decide what to put on disk, for how long and how to replace subsets on disk
     - Users have to use the computer centers
     - Duplicate copies at different locations increase management difficulty
     - Many users build their own private subsets

Subset Hell

Consider the case of a physicist with a 20-node cluster who wants to try out a brilliant idea

- Needs to copy data from central storage sites to the cluster
  1. Find the locations of all needed files
  2. Decide how to split files among machines in the cluster
  3. Perform transfer, retry if transfer failed for whatever reason
  4. Run analysis code
  5. Build own subsets for future analyses
  6. When disks are full, decide what to remove, repeat 3—6
- In this process, the actual time to run analysis code might be 10% or less!
- To try another idea, may have to repeat all steps

Let Programs Do It

- Changing from "let professionals do it" to "let programs do it"
- A number of similar efforts
  - Most can only deal with files on disk
  - GriPhyN Virtual Data Toolkit (Livny and Roy, Wisconsin)
  - PROOF, a parallel extension to ROOT (Ballintijn et al., MIT)
  - DIAL, Distributed Interactive Analysis of Large datasets (Adams, Brookhaven)
  - Grid Enabled Analysis, a collections of projects (Bunn, et al, CalTech)
  - JAS, …
- What is unique in our approach
  - Index all events to enable direct access to selected events
  - Retrieve files from mass storage systems
  - Automatic garbage collection

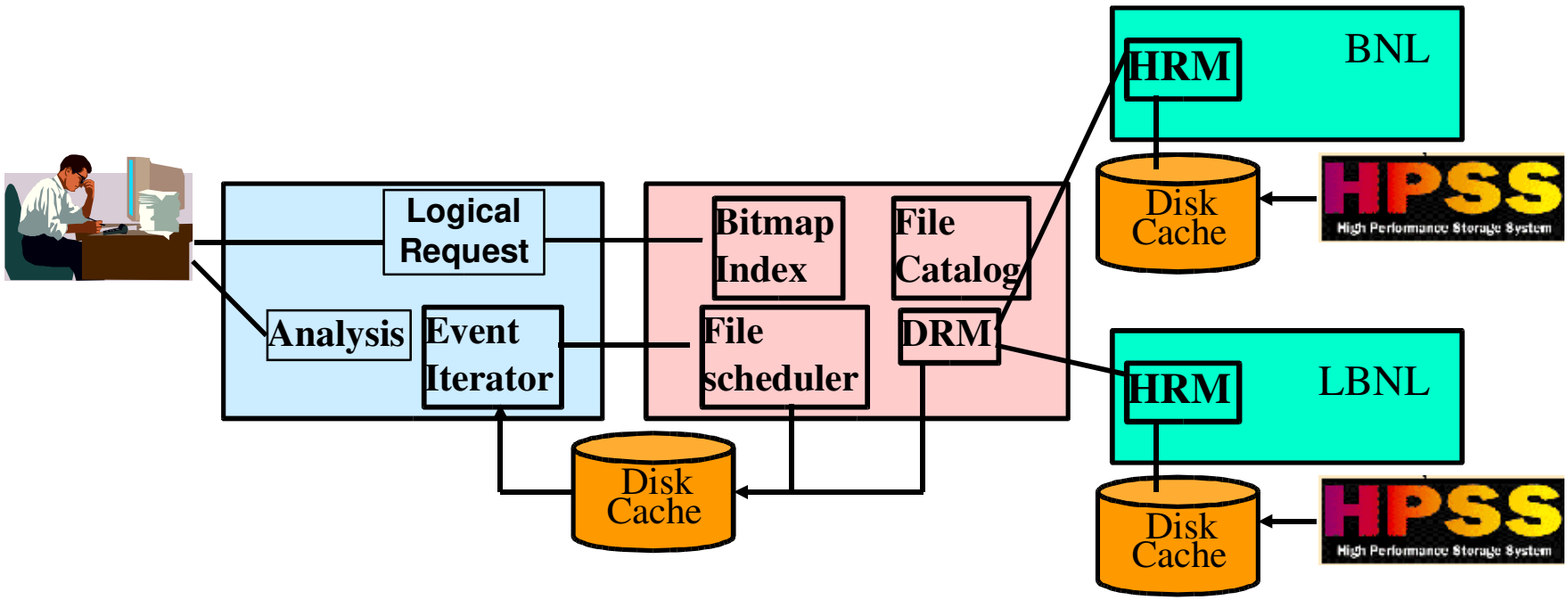
Features of Grid Collector

- Transparent object access
  - No need for analysts to manage files or disk space
  - No need for analysts to access remote mass storage systems
- Select objects based on tag values
  - E.g., production=P03ia & numberOfPrimaryTracks>200
- Improve analysis system's throughput by
  - Eliminating the need to read all objects in a file
  - Providing optimized disk space management and automatic garbage collection
  - Automating the retrieval of files from remote storage systems
- Interactive analysis of data distributed on the GRID
  - Providing quick partial answers
  - Enabling users to transparently share files in disk caches

For all users, it is an efficient filter

Schematics of Grid Collector

- **Users specify what events are desired as a logical request**
- **The selected events are delivered one at a time to analysis code**



The Building Blocks

- Bitmap Index
  - Indexes each event
  - Efficient for partial range queries
- Storage Resource Managers
  - Manages disk cache (DRM)
  - Automatic retrieval of needed files from the Grid
  - Automatic retrieval from HPSS (HRM)
- File Scheduler
  - Coordinates file accesses
- File Catalog
  - Provides location information about files
- Index Feeder
  - Digests ROOT files to extract information about events (tags)
- Event Iterator
  - Feeds events to analysis code in a stream

Using Grid Collector

- Existing practice
  - Specify a list of files or directories containing the desired events
  - Analyze all events in the files
    - Reading more events than needed
  - Files have to be on disk before analysis
    - User has to manage the files and space
    - All files have to be present at the same time
- Using Grid Collector
  - Specify the conditions characterizing the desired events, such as “production=P03ia & numberOfPrimaryTracks>=200”
  - Analyze only events satisfying the conditions
    - Bitmap index provides keys to access only the selected events
  - Files are retrieved and managed by the Grid Collector
    - User does not have to know about the files
    - Files are retrieved in a stream, reducing the disk space required

Use Case – I

- Using a sample analysis script called doEvents.C
- Analyze first 100 events from production P03ia with 200 or more primary tracks
  - .x doEvents.C(100, “where production=P03ia & numberOfPrimaryTracks>=200”)
- To analyze all events, set the first argument to a negative integer
- To try different conditions without analyzing them, a separate command is available

- Without using Grid Collector
  - .x doEvents.C(100, “/star/data10/gc/WRK/cache/*.event.root”)
  - Analyze first 100 events in the files
  - Files need to be on disk
  - Need additional code to skip unwanted events

Use Case – II

Creating your own script to use the Grid Collector

- Load StGridCollector library
- Create an object of type StGridCollector
- Initialize the object with a select statement
- Pass the object to StIOMaker just like a StFile object
- The rest of the code is exactly the same as using StFile

In doEvents.C, change the line “setFiles = new StFile(fileList);” into

```
3.  if (strncmp(fileList[0], “where ”, 6) == 0) { // use Grid Collector
4.          gSystem->Load("StGridCollector");
5.          setFiles = StGridCollector::Create(fileList[0]);
6.  } else { // assume fileList to be a normal file list
7.          setFiles = new StFile(fileList);
8.  }
```

Syntax of Select Statement

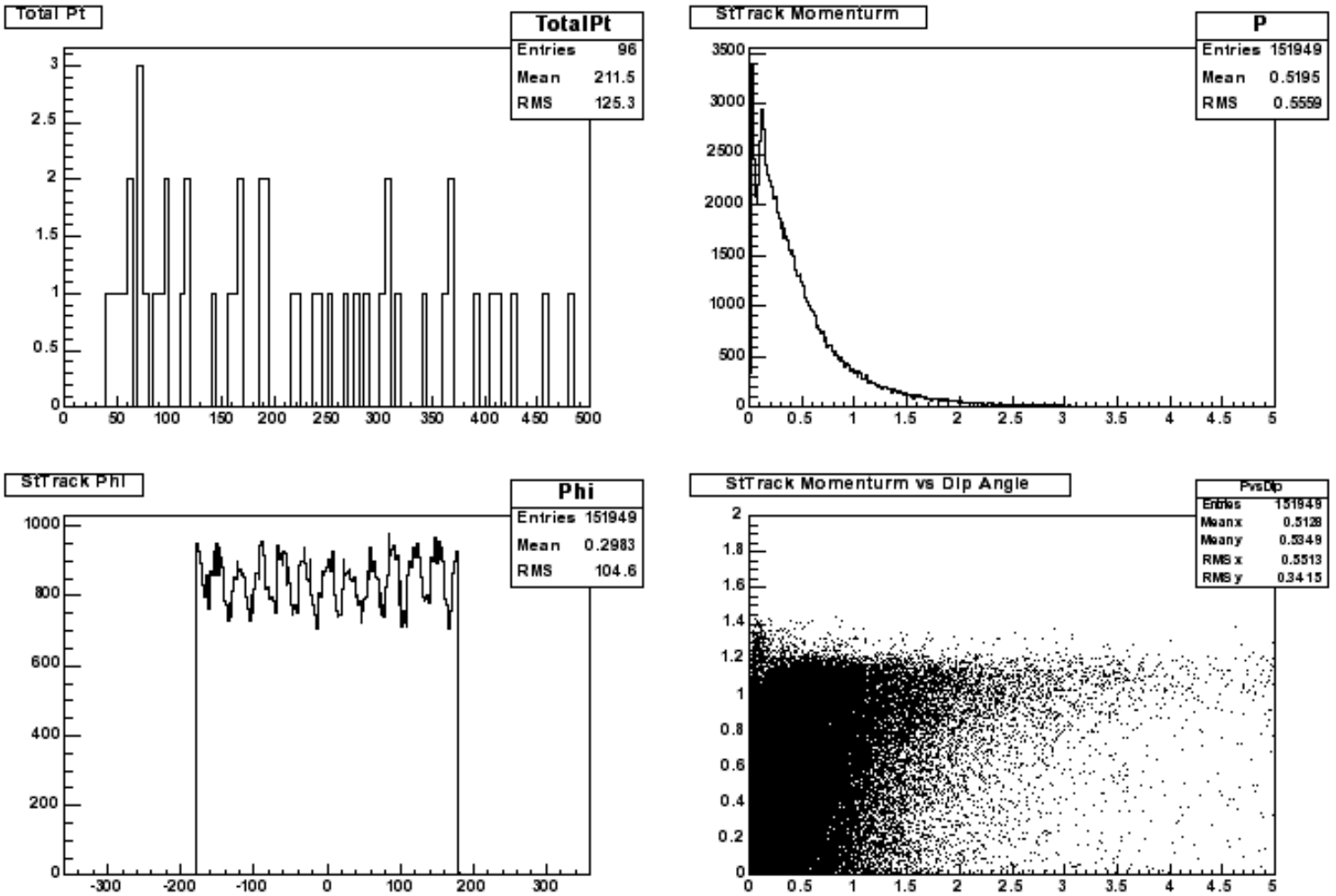
- To initialize a StGridCollector object one may use a select statement
- SELECT … FROM … WHERE …
  - A simplified version SQL select statement
- Select clause indicates the type of files, event, MuDst, …
  - If omitted, assumed to be “SELECT event”
- From clause indicates the name of dataset
  - If omitted, assumed to be “FROM *”
- Where clause indicates the conditions
  - Join simple conditions together with AND, OR, XOR, NOT
  - A simple condition is an equation (A = 5), an inequality (A > 5), or a range (5 <= A < 10)
  - The attribute name can be an arithmetic expression, 5*sqrt(A*A+B*B)
  - String attributes can only appear in equations

Alternative Initialization Scheme

- An alternative to use the select statement is to use flags and arguments
- The following two lines are equivalent
  - .x doEvents.C(100, “where production=P03ia & numberOfPrimaryTracks>=200”)
  - .x doEvents.C(100, “GC -q ‘production=P03ia & numberOfPrimaryTracks>=200’”)
- Second form used to specify addition operations
  - Processing a query established elsewhere (-t token)
  - Access events of specified run numbers and event numbers (-i file-containing-the-numbers)

A Simple Run

root4star -b -q doEvents.C'(100, “where Production=P02gc and BfieldReversedFullField and chargedMultiplicity>100”)'



Out Of Subset Hell ?

Back to the case of our professor with a 20-node cluster, what does he do with the Grid Collector ?

- External servers needed: File Catalog, Grid Collector server, HRM – only need information about them
- Local software required: STAR with Grid Collector, ROOT, DRM, Globus, ORBacus
- Local server: DRM (plus a piece of disk for it to store files)
- A job on one node
  - root4star –q –b doEvents.C‘(-1, “select …”, “evout”)’

6. A large job on multiple nodes
   - Estimate the job with a command line tool, obtain tokens
   - Start multiple jobs with “GC –t token”

Status and Future Plans

- Current state
  - Grid Collector handles event files
  - Populating the indices now
- Future plans
  - Handle MuDst files (March 2004, John, need help)
  - Speed up the index building process (March 2004, Wei-Ming)
  - New tags, e.g., centrality (? ?)
  - Parallel analyses for large jobs (March 2004, John)
  - Analyze events in a specified order (December 2004, John)
  - Make it into a Grid-enabled service (2005, John)
- Contact information
  - John Wu <John.Wu@nersc.gov>
  - Wei-Ming Zhang <zhang@hpacq.kent.edu>
  - Jerome Lauret <jeromel@bnl.gov>