# Data Mining and Physics Analysis Tools

**Background Notes for**
**ATLAS Analysis Tools Workshop**

**Geneva**

**20 May 1999**

**David Malon**

# Data Mining?

I am never quite sure what physicists mean when they speak of data mining.

In the commercial and data mining research worlds, data mining is often associated with specific kinds of knowledge discovery, and with specific techniques.

# Mining for Association Rules

- **Standard example:**

  **People who buy bananas are likely to buy milk.**

- **Physics example:**

  **Higgs events occur primarily in data taken on Mondays.**

**We may learn interesting things by mining for association rules (though physics may not be one of them).**

# Standard Data Mining Questions

- **What spending patterns are indicative of (predictive of) credit card fraud?**
- **What properties characterize credit card customers who are good credit risks?**
  - To whom should you mail your solicitations?
  - Which applicant profiles should you accept?
    - “Yes” to the student applicant with no income who says she is majoring in pre-medicine?
    - “No” to the student applicant with no income who says he is majoring in physics?
- **(Construct your own physics parallels.)**

# Standard Techniques in “Siftware”

- **Classification** -- for building a classification model
  Approach: **Multiple** | **Decision tree** | **Rules** | **Neural network** | **Bayesian** | **Other (Rough sets, Genetic, Nearest Neighbour, ...)**
- **Clustering** - for finding clusters or segments
- **Statistics, Estimation and Regression**
- **Links and Associations** - for finding links, dependency networks, and associations
- **Sequential Patterns** - tools for finding sequential patterns
- **Visualization** - scientific and discovery-oriented visualization

# Other Kinds of Siftware Tools

- Text and Web Mining
- Deviation and Fraud Detection
- Reporting and Summarization
- Data Transformation and Cleaning
- OLAP and Dimensional Analysis

# Some Definitions of Data Mining

## data mining

- **The extraction of hidden predictive information from large databases**

## data mining

- **An information extraction activity whose goal is to discover hidden facts contained in databases**. Using a combination of machine learning, statistical analysis, modeling techniques and database technology, data mining finds patterns and subtle relationships in data and infers rules that allow the prediction of future results. Typical applications include market segmentation, customer profiling, fraud detection, evaluation of retail promotions, and credit risk analysis.

# Some Definitions of Data Mining

- Data mining
  - The term data mining is somewhat overloaded. It sometimes refers to the whole process of knowledge discovery and sometimes to the specific machine learning phase.
- Knowledge discovery
  - The non-trivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data. This is the definition used in ``Advances in Knowledge Discovery and Data Mining,'' 1996, by Fayyad, Piatetsky-Shapiro, and Smyth.

# Common Threads

- The essential characteristic is knowledge discovery, not information retrieval.
- There is commonly an emphasis upon discovery of patterns and predictive models.

# What do physicists mean by data mining?

- Some people associate it with random access to large amounts of data
- Others associate it with computationally intensive event-by-event analysis
- By some definitions, almost all physics data analysis is data mining

In any case, off-the-shelf mining tools are unlikely to compete with codes that reflect deep physics knowledge,

but

it takes tremendous hubris to assume that we can learn nothing from them.

# Data architectures to support physics data mining

Just an impression:

Most people begin with the idea of physicists doing analysis based upon disk-resident condensed views of events, DYNAMICALLY navigating to data on tertiary storage when they find interesting events.

As the time for deployment draws near, people responsible for delivering reconstruction and analysis architectures want to DISALLOW (or at least strongly discourage) pure random access to data on tape by individual physicists.

# Tag Databases

- Underlying notion is that a subset of event attributes are used to do event selection for deeper analysis
- Such attributes constitute an event’s “tag”
- Tags are assumed disk-resident
- Some people organize tags into a tag database
- Others build indices into primary event store; tags are simply the indexed attributes
- Some do both

# DOE Grand Challenge Project for High Energy and Nuclear Physics

- A collaboration among Department of Energy laboratories and researchers at several universities
- Began mid-1997
- Aim is to provide tools for large-scale (hundreds to thousands of terabytes annually) management and analysis of experimental physics data
- RHIC data system prototype in 1998, production system in 1999 are primary targets
- Primary deliverable to date:  order-optimized, multiuser prefetch architecture for data on tertiary storage, with query estimation capabilities

# General Motivation

- Physics databases may contain hundreds or thousands of terabytes of data and span thousands of files
- Many of those files will reside on tape
- A few keystrokes may be the difference between a query that returns 100 events and a query that returns 100,000,000 events distributed over thousands of files on thousands of tapes
- Such large queries might be entirely appropriate, but useful tools might:
  - help users understand the scope and ramifications of their queries *before* they execute
  - optimize access to data within a given query, and among concurrent queries

# Query Estimator

- For a given query, estimator returns:
  - number of objects that satisfy a selection query
  - number of files touched
  - estimate (currently crude) of total retrieval time based upon what data are on tape, what are on disk, and (eventually) the concurrent query load
- Two options in the Grand Challenge architecture:
  - Quick Estimate: consults an optimized (likely bit-sliced) index in memory for approximate answers (uses attribute binning),
  - Full Estimate: consults a “Tag” database or full index for more precise answers
  - User decides whether to proceed on the basis of estimates

# What comprises a query?

- A selection predicate OR a collection of event references
- native GCA query language (“RangeQL”) allows boolean combinations of range selections on indexed attributes

  (1800<num_Pion_p<2000) AND (2000>num_Pion_n)

- current LocalDbResources implementation also supports ObjectivityQL queries (selection predicates on tag data members)

  _num_Pion_zero>0.585*(_num_Pion_p+_num_Pion_n)

- the latter uses collections-as-queries support: builds an in-memory collection from results returned by Objectivity predicate scan

# Collections as Queries

- Example
  - User input: iterator over a personal or collaboration-wide collection of ooRefs (if Objectivity) to interesting objects
  - Estimator output: collection cardinality, number of files involved, time estimate
  - GCA value added: optimized iteration over the same collection
- Current CORBA implementation, though, passes collections of object references, rather than iterators, as queries

# Storage Manager

- Because queries pass through an execution interface, a Storage Manager can determine all the files that all concurrent queries will need
- Maintains knowledge of what data are on disk and what are on tape
- Prefetches--optimizes tape access within a query, and among concurrent queries
  - notices if a query requires two databases on the same tape
  - more likely to deliver a database that will service several concurrent queries
  - a separate Policy Module (still in its infancy) informs prefetch decisions

# Order Optimized Iterator

- A query execution request returns a query token that seeds an iterator, which has the same interface as an STL iterator or an ODMG d_Iterator<T> over a set.
- The *order* in which elements are returned, though, is indeterminate. The behavior is as though object references had been sorted by database, and the qualifying databases had been sorted by access latency.
- The iterator talks to the Storage Manager, and gets a (sub)list of event references corresponding to a disk-cached database.
- When the user invokes, e.g., iter.next(ObjectRef), a reference is returned from this sublist; when the sublist is exhausted, iter.next causes the iterator to get a new sublist from the Storage Manager (which has, presumably, prefetched another database).

# Notes on Order Optimized Iteration

- **Note that order-optimized iteration is identical to unoptimized iteration--it just runs faster (we hope)**
- **Designed for “for each” analysis--when the user wants each qualifying object, but doesn’t care about order**
- **While the architecture is intended to be general, the current implementation uses Objectivity, where an ObjectRef is really an ooRef(Event)**
- **Straightforward to parallelize iteration**

# Parallel Iteration

- Broadcast the QueryToken to worker processes running in SPMD (Same Program, Multiple Data) mode
- Use token to initialize order-optimized iterators in each process
- Parallel processes all talk to the same Storage Manager
- Storage Manager gives next sublist of ObjectRefs to whichever process asks next--guarantees that no two iterators deliver the same ObjectRefs
- Have used:
  - poor man's parallelism (multiple rsh commands each with the same query token as an argument)
  - a portable implementation of MPI (Message-Passing Interface)--standard in the large-scale parallel computing world

# GCA Components and CORBA

- CORBA-based connections
  - using Orbix on Suns, Omnibroker (for now) on Linux platforms and Suns
- CORBA components
  - Query Factory and Query objects
  - Query Estimator
  - Query Monitor (Storage Manager functions in earlier slides)
  - Cache Manager (HPSS connections and multi-query cache management)
- CORBA clients
  - physics analysis codes (implicit; users need not know about CORBA)
  - Order Optimized Iterator (may become CORBA server for callback)
  - Policy Manager/Module (hidden)

# GCA Components and Objectivity

- Architecture is intended to be general, but current implementation uses Objectivity (ROOT, too, very soon)
- STAR Tag database and event data delivered via GCA are in Objectivity
- Event references, while passed opaquely through the architecture via CORBA, are really Objectivity ooRefs
- Cache Manager uses pftp behind Objectivity's back to deliver databases to Objectivity AMS-accessible disk cache
  - coordinating with Andy Hanushevsky's (SLAC) Objectivity/HPSS work
- Output collections of events can be saved in Objectivity as collections of ooRefs
- Can support Objectivity predicate scan and query estimation without external GCA components

# Physics Analysis with the GCA Architecture

**If you can navigate the STAR persistent event data model:**

1. Write a routine to do analysis on a single event; signature is

   void usercode(d_Ref_Any& current);
2. Link with a precompiled driver
3. Invoke the executable with an optional query string on the command line

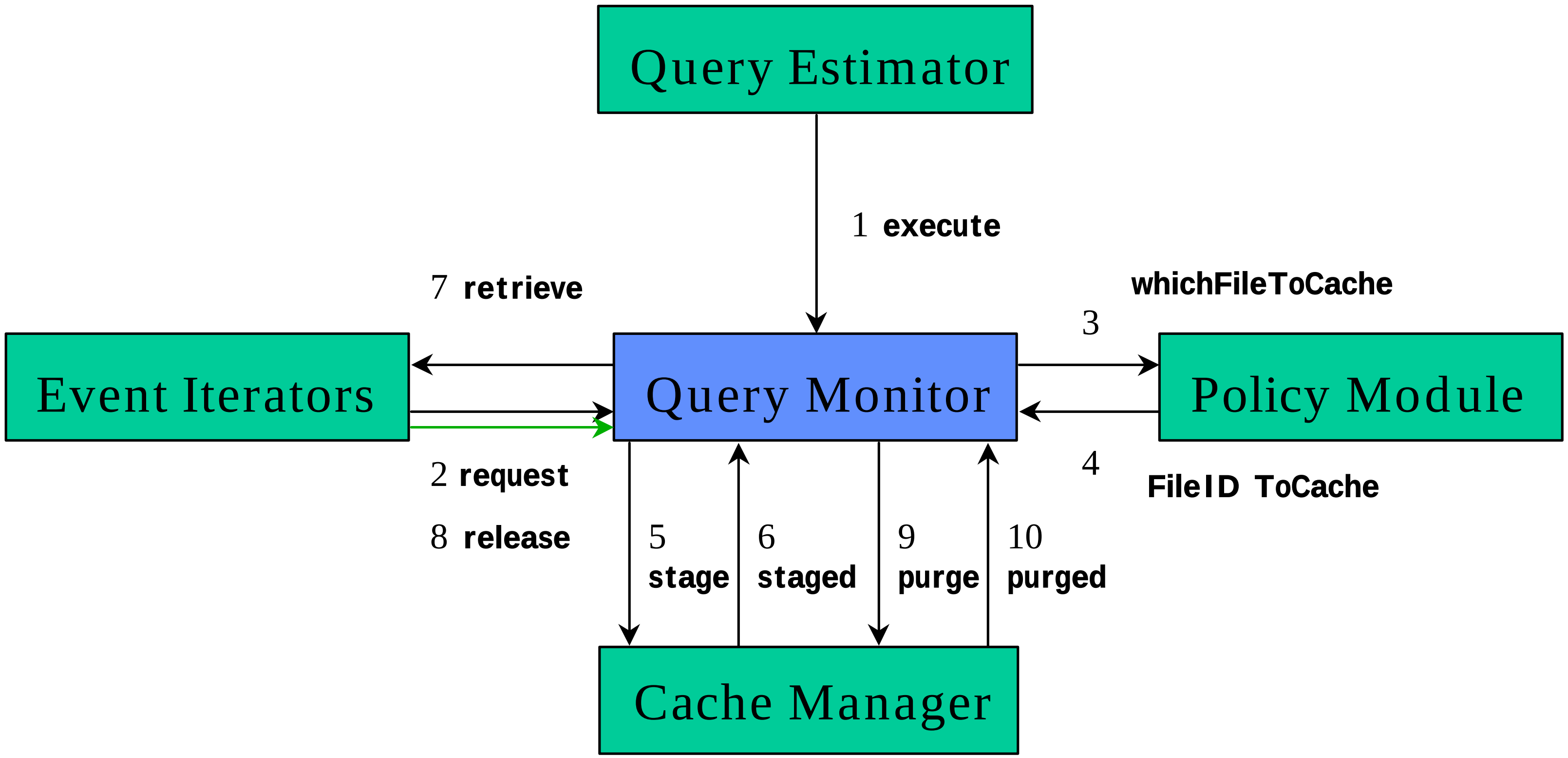
You need know nothing about the GCA architecture and CORBA.

Note: d_Ref_Any is used because the architecture does not know YOUR data model.

For most physicists, signature is

void usercode(STAR_TransientEvent& current);

(or a ROOT equivalent)


Query Estimator
1 execute
7 retrieve
whichFileToCache
3
Event Iterators
Query Monitor
Policy Module
2 request
8 release
4
FileID ToCache
5
stage
6
staged
9
purge
10
purged
Cache Manager

# Less Ambitious Models: Data Trains/Carousels

**SYSTEM:**

- **Tape drives run continuously, streaming all event data to disk cache**
- **Disk cache is FIFO: arriving data displaces oldest data in cache**
- **When all events have been read, do it again**

**USER:**

- **Submit job whenever you want, noting where in the event stream you arrive**
- **Try to keep up with tape speeds in your analysis; if you fail, you can stick around through the next data cycle**
- **Run until your first event arrives again**

# Data Trains/Carousels

- Obvious possible strategy when analyses that look at complete event data for all events are common, but
- Also a potential solution when hundreds of physicists are requiring simultaneous random tape-resident data

# Datastores for Physics Analysis

- **Predominant plan for near-term experiments seems to be to support most analysis out of microDSTs (or similar reduced, largely disk-resident, datasets**
- **Example: STAR strategy**
  - standard production jobs, one for each physics working group, produce microDSTs from reconstructed data
  - Aside: microDSTs are ROOT files; analysis is done with ROOT

# Physicists and Canonical Data Mining

- Is anyone applying standard data mining techniques to HEP data?

  I don't know.

- As a spare-time project, David Zimmerman (LBNL) and I are looking at applying rough set techniques to HENP data.
- Rough sets: fuzzy classifiers/associators, with connections to implicit dimensionality
- Example: Customers rate 100 cars. Some are considered excellent. Now look at a dozen attributes of cars (color, size, horsepower, price, …). Can you predict (roughly) from a subset of those attributes which cars are excellent? Can you find a minimal subset?

# Rough Sets in Physics Data Mining

- PHYSICS SCENARIO (posed by David Zimmerman):
  - By grueling and computationally intensive analysis, you have identified a collection of interesting events.
  - Now look at the tag database. Is there a selection specification based upon those tag attributes that would have (roughly) delivered these events?
- More talk than action thus far (no results). The problem with spare-time projects is that spare time is nonexistent.