Big Data Processing with Less Work and Less Code

Parsing Semi-Structured and Free Form Text

Richard Taylor

LexisNexis

Welcome!

HPCC Systems
http://hpccsystems.com

- HPCC Platform Overview
- Getting Started
  - Installations
  - Data Acquisition
  - Data Definition
- Parsing Semi-Structured Data
- Parsing Unstructured Data
- Post-Processing Parse Results

Primary Goals of this Class

HPCC Systems
http://hpccsystems.com

- Familiarity with the HPCC Platform
- Familiarity with the ECL (Enterprise Control Language) Programming tools:
  - ECL IDE
  - ECL Watch
- Familiarity with Basic ECL Concepts and Syntax
- Familiarity with ECL's Parsing Technology

Where We're Going

HPCC Systems
http://hpccsystems.com

- Quick Demo:
  - Parsing Free Form Text (BWR_QuickDemo)

Why Does HPCC Exist?

HPCC Systems
http://hpccsystems.com

- It was NOT developed with the idea of selling the technology to anybody else!
- It was all created only to solve some of the data-handling problems that we encountered as we were developing our products.

How Was HPCC Developed?

HPCC Systems
http://hpccsystems.com

- **1999**
  - **Our Problem: INDEXes**
    We had data files with literally hundreds of INDEXes into them.
  - **Our Solution: HOLe**
    Our first massively parallel processing platform, HOLe (now deprecated and out of use for many years), eliminated any need for INDEXes by making every query a full table scan against all the data, which was all loaded into memory, and used SQL for its queries.

How Was HPCC Developed?

HPCC Systems
http://hpccsystems.com

- **2000**
  - **Our Problem: SQL**
    The kind of really complex queries we needed to implement would have taken too much work to develop in SQL.
  - **Our Solution: ECL**
    A new data-specific programming language, tightly coupled to the platform, terse and very expressive, getting a lot of work done with very little actual code.

How Was HPCC Developed?

HPCC Systems
http://hpccsystems.com

- **2001**
  - **Our Problem: INDEX building**
    Building multi-billion row INDEXes was time-consuming.
  - **Our Solution: Thor**
    Our second massively parallel processing platform, Thor, was designed specifically to build INDEXes on massive amounts of data, and to do all the standard ETL work around getting raw data into product form.

How Was HPCC Developed?

HPCC Systems
http://hpccsystems.com

- **2002**
  - **Our Problem: Using INDEXes**
    Building INDEXes on Thor and then moving them to another platform for use was time-consuming. Maintaining separate platforms for data production versus data delivery was inefficient.
  - **Our Solution: Roxie**
    Our third massively parallel processing platform, Roxie, was designed specifically to deliver data to end users using the same technology base as the tools used to build the products.

The Result Of All That Development?

HPCC Systems
http://hpccsystems.com

- **HPCC Systems**

A single, fully-integrated platform supporting the entire life cycle of Big Data product development:

- Raw Data Ingest – Thor
- Data Transformation to Product – Thor
- End-user Query Development – Thor
- End-user Query Delivery – Roxie

The Complete Big Data Value Chain

HPCC Systems
http://hpccsystems.com

Collection | Ingestion | Discovery & Cleansing | Integration | Analysis | Delivery

Collection – collecting structured, unstructured and semi-structured data

Ingestion – consuming vast amounts of data including extraction, transforming and loading

Discovery & Cleansing - clean up, formatting and statistical analysis of the data

Integration – linking, indexing and data fusion

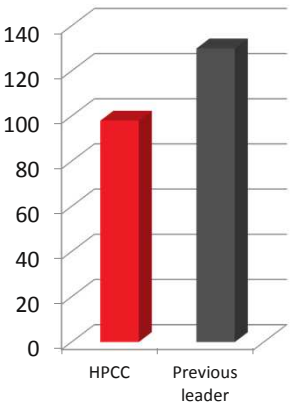
Analysis – statistics and machine learning

Delivery – querying, visualization, and redundancy, enterprise-class availability

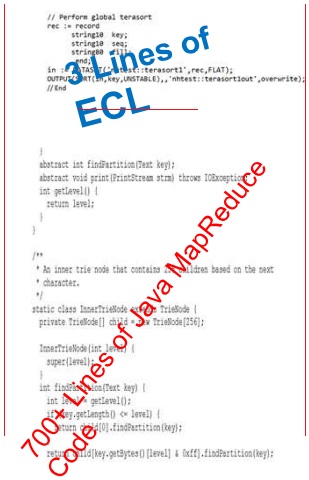
Terasort Benchmark Results (less is better)

HPCC Systems
http://hpccsystems.com

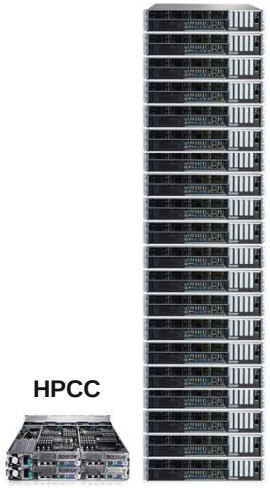
Execution Time (seconds)



Productivity



Space/Cost



Terasort Benchmark Results (less is better)

HPCC Systems
http://hpccsystems.com

http://www.sgi.com/company_info/newsroom/press_releases/2011/october/hadoop.html

"Results achieved in **September, 2011** show that a **20-node SGI Hadoop cluster** comprised of **SGI**® Rackable™ C2005-TY6 half-depth servers with Intel® Xeon® processor E5630 series, 48GB of memory, and 4x 1TB SATA HDDs running on Cloudera CDH3 took only **130 seconds to complete a Terasort with a job size of 100GB**."

---

http://hpccsystems.com/about-us/press_center/press_releases/breaks-world-record-terasort-benchmark-121211

"Results achieved in **December 2011** show that an **HPCC Systems four node Thor cluster took only 98 seconds to complete a Terasort with a job size of 100 gigabytes** (GB) on a cluster five times smaller than Hadoop. The HPCC Systems four node cluster was comprised of one (1) Dell PowerEdge C6100 2U server with Intel® Xeon® processors E5675 series, 48GB of memory, and 6 x 146GB SAS HDD's. The Dell C6100 houses four nodes inside the 2U enclosure."

HPCC Platform

HPCC Systems
http://hpccsystems.com

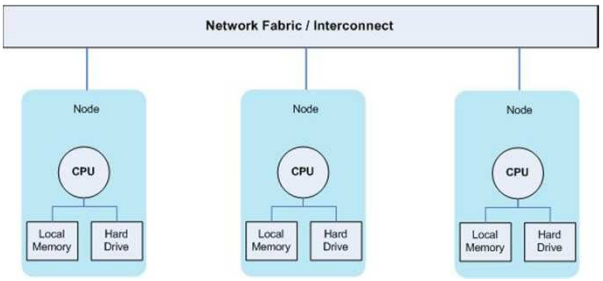
There are two types of clusters in the HPCC:

- Data Refinery (THOR) – Processes every one of billions of records in order to create billions of "improved" records – runs one job at a time.
- Rapid Data Delivery Engine (ROXIE) – Searches quickly for a particular record or set of records – handles thousands of concurrent transactions per second.
- These are tightly coupled to the infrastructure that supports their operation, and the ECL programming language that defines the work done on them

HPCC Hardware

HPCC Systems
http://hpccsystems.com

- Clusters of commercial off-the-shelf components (COTS). Components are ideally homogeneous (all processing/disk storage components same) and the system is tightly coupled.
- Nodes are managed *en masse* instead of individually, which allows coordinated processing like global sorts (unlike Grid systems).



Thor Cluster

HPCC Systems
http://hpccsystems.com

- **Brute force:** Thor operates on massive amounts of data where datasets typically contain billions of records
- **Open Data Model:** The data model is defined by the user, not constrained by the limitations of a strict key-value paradigm
- **Scalable:** Horizontally linear scalability provides room to accommodate future data and performance growth
- **Truly parallel:** Datagraph Nodes can be processed in parallel as data seamlessly flows through them, effectively avoiding the well-known "long tail problem", resulting in higher and predictable performance.
- **Powerful optimizer:** The HPCC optimizer ensures submitted ECL code executes at the maximum possible speed for the underlying hardware. Advanced techniques such as lazy execution and code reordering are thoroughly utilized to maximize performance

Roxie Cluster

HPCC Systems
http://hpccsystems.com

- **Low latency:** Data queries typically complete sub-second
- **Not a key-value store:** Roxie is not limited by the constraints of key-value data stores, allowing for complex queries, multi-key retrieval, fuzzy matching and more
- **Highly available:** Roxie operates in critical environments under the most rigorous service level requirements
- **Scalable:** Horizontally linear scalability provides room to accommodate future data and performance growth
- **Highly concurrent:** In a typical environment, thousands of concurrent clients can be simultaneously executing transactions on the same Roxie system
- **Redundant:** A shared-nothing architecture with no single point of failure provides extreme fault tolerance

HPCC Platform

HPCC Systems
http://hpccsystems.com

- **Batteries included:** All components create a consistent and homogeneous platform
- **Over 10 years of experience:** The HPCC platform is the technology underpinning LexisNexis data offerings – its development began in 1999
- **Few moving parts:** HPCC is an integrated solution extending across the entire data lifecycle, from data ingest and transformation to data delivery – no third party tools needed
- **Multiple data types:** Supported out of the box, including fixed and variable length delimited records and XML
- **ECL inside:** One language to describe both: the data transformations in Thor and data delivery strategies in Roxie. Solutions to complex data problems are expressed easily and directly in terms of high level ECL primitives.
- **Consistent tools:** Thor and Roxie share the same set of tools, which provides consistency across the platform.

Data on the HPCC

HPCC Systems
http://hpccsystems.com

- **Open Data Model:** The data model is defined by the user, as standard files and fields (tables and columns)
- **Simple:** Solutions to complex data problems can be expressed easily and directly in terms of high level ECL primitives
- **Implicitly parallel:** Data is always in distributed datasets whose parts are managed by the DFU, eliminating the need for programmers to manage the complexity of working with distributed datasets

Data on the HPCC

HPCC Systems
http://hpccsystems.com

- Data is stored in ISAM Files
- Native support for:
  - Flat files, with fixed or variable-length records
  - CSV-type files (any delimiters may be used)
  - XML datasets
- Each Record is always whole and complete on a single node
- A Record may have as many fields as needed
- Indexes are always LZW compressed and may contain “payload” fields in addition to search terms

What is ECL?

HPCC Systems
http://hpccsystems.com

- **Declarative programming language:** Describes what needs to be done and not how to do it
- **Powerful:** Unlike Java, high level primitives such as JOIN, TRANSFORM, PROJECT, SORT, DISTRIBUTE, MAP, etc. are available. Higher level code means fewer programmers and shorter time to deliver complete projects
- **Extensible:** As new definitions are created, they become primitives that other programmers can use
- **Implicitly parallel:** Parallelism is built into the underlying platform. The programmer need not be concerned with it
- **Maintainable:** A High level programming language, no side effects and defintion encapsulation provide for more succinct, reliable and easier to troubleshoot code
- **Complete:** Unlike Pig and Hive, ECL provides for a complete programming paradigm.
- **Homogeneous:** One language to express data algorithms across the entire HPCC platform, including data ETL and delivery.

Getting Started

HPCC Systems
http://hpccsystems.com

- **Install:**

1. VMware Player:
   E:\Installs\Vmware
   or Oracle's VirtualBox
   https://www.virtualbox.org/wiki/Downloads

2. ECL IDE:
   E:\Installs\ECL_IDE

Getting Started

HPCC Systems
http://hpccsystems.com

- **Run:**

1. Launch your VM player.
2. Import the HPCC Virtual Machine .ova file
3. Note the IP next to the **IP Address:** prompt at the top of the VM.

   This IP address is the key to allowing the HPCC client tools to access the environment.

Getting Started

HPCC Systems
http://hpccsystems.com

- **Run:**

3. Open a browser and go to the ECL Watch page URL referenced in the paragraph following the IP Address.

   This web page is a key resource to have open while working with HPCC. ECL Watch allows you to monitor the health of the environment, browse your datasets and workunits, and most importantly, to get data into and out of the HPCC environment.

Getting Started

HPCC Systems
http://hpccsystems.com

- **Run:**

4. Open the ECLIDE.exe application and:
   1. In the **Preferences** dialog, enter the previously noted IP (only) into the **Server** entry control.
   2. Go to the **Compiler** tab and press the **Add** button below the **ECL Folders** list.
   3. Select the E:\Code folder and press **OK**.
   4. Press **OK** to get back to the Login dialog,
   5. Press **OK** to login.

Acquiring Data

HPCC Systems
http://hpccsystems.com

1. Task-switch to the **ECL Watch** page in your browser
2. Click the **Upload/download Files** link in the **DFU Files** section of the menu on the left side of the page
3. Press the **Choose File** button and select the **E:\data\actresses.list** file from your thumb drive
4. Press the **Upload Now** button (big file may take awhile)
5. Press the **Choose File** button and select the **E:\data\KJV.txt** file from your thumb drive
6. Press the **Upload Now** button

Acquiring Data

HPCC Systems
http://hpccsystems.com

1. Click the **Spray Delimited** link in the **DFU Files** section of the menu on the left side of the page
2. Press the **Choose File** button and select the **actresses.list** file from your Landing Zone
3. Delete the content of the **Separator** and **Quote** entry controls and check the **No Separator** box
4. Type **CTS::actresses** into the **Label** entry control
5. Press the **Submit** button
6. Click the **View Progress** link to watch the process

Acquiring Data

HPCC Systems
http://hpccsystems.com

1. Click the **Spray Delimited** link in the **DFU Files** section of the menu on the left side of the page
2. Press the **Choose File** button and select the **KJV.txt** file from your Landing Zone
3. Delete the content of the **Separator** and **Quote** entry controls and check the **No Separator** box
4. Type **CTS::KJV** into the **Label** entry control
5. Press the **Submit** button
6. Click the **View Progress** link to watch the process

Defining the Files

HPCC Systems
http://hpccsystems.com

1. Task-switch to the **ECL IDE**
2. Open the **Actresses** ECL code file
3. Examine the code that defines the file
   1. The RECORD structure
   2. The DATASET declaration

Defining the Files

HPCC Systems
http://hpccsystems.com

1. Look at the defined data
   1. Select your Thor cluster as the **Target**
   2. Press the **Submit** button

Defining the Files

HPCC Systems
http://hpccsystems.com

1. Open the **KJV** ECL code file
2. Examine the code that defines the file
   1. The RECORD structure
   2. The DATASET declaration
3. Look at the defined data
   1. Select your Thor cluster as the **Target**
   2. Press the **Submit** button

Parsing Semi-Structured Text

HPCC Systems
http://hpccsystems.com

1. Open the **ParseActress** ECL code file
2. Examine the code
   1. PATTERN definitions
   2. The PARSE function

Overview of Natural Language Parsing

HPCC Systems
http://hpccsystems.com

Natural Language Parsing is accomplished in ECL by combining pattern definitions with an output RECORD structure specifically designed to receive the parsed values, then using the PARSE function to perform the operation.

Pattern definitions are used to detect "interesting" text within the data. Just as with all other ECL definitions, these patterns typically define specific parsing elements and may be combined to form more complex patterns, tokens, and rules.

The output RECORD structure (or TRANSFORM function) defines the format of the resulting recordset. It contains specific pattern matching functions that return the "interesting" text, its length or position.

The PARSE function implements the parsing operation. It returns a recordset that may then be post-processed as needed using standard ECL syntax, or simply output.

Parsing Semi-Structured Text

HPCC Systems
http://hpccsystems.com

1. Look at the parsed result
   1. Select your Thor cluster as the **Target**
   2. Press the **Submit** button

Parsing Unstructured Text

HPCC Systems
http://hpccsystems.com

1. Open the **ParseKJV** ECL code file
2. Examine the code
   1. The TABLE function
   2. The TRANSFORM function
   3. The ROLLUP function
   4. The SORT function

Parsing Unstructured Text

HPCC Systems
http://hpccsystems.com

1. Look at the parsed result
   1. Select your Thor cluster as the **Target**
   2. Press the **Submit** button

Post-Processing Parse Results

HPCC Systems
http://hpccsystems.com

1. Open the **BibleNames** ECL code file
2. Examine the code
   1. The FUNCTION structure
   2. The IF function
   3. The SET function
   4. Inline DATASET definitions
   5. The DEDUP function

Post-Processing Parse Results

HPCC Systems
http://hpccsystems.com

1. Look at the result
   1. Select your Thor cluster as the **Target**
   2. Press the **Submit** button

More Post-Processing

HPCC Systems
http://hpccsystems.com

1. Open the **MatchNames** ECL code file
2. Examine the code
   1. The JOIN function

More Post-Processing

HPCC Systems
http://hpccsystems.com

1. Look at the result
   1. Select your Thor cluster as the **Target**
   2. Press the **Submit** button

And Even More Post-Processing

HPCC Systems
http://hpccsystems.com

1. Open the **SecondDegree** ECL code file
2. Examine the code
   1. The TRIM function

And Even More Post-Processing

HPCC Systems
http://hpccsystems.com

1. Look at the result
   1. Select your Thor cluster as the **Target**
   2. Press the **Submit** button

One More Before You Go

HPCC Systems
http://hpccsystems.com

1. Open the **ParseReplace** ECL code file
2. Examine the code
   1. The PROJECT function
   2. The CASE function
   3. The CHOOSE function
   4. The EVALUATE function

One More Before You Go

HPCC Systems
http://hpccsystems.com

1. Look at the result
   1. Select your Thor cluster as the **Target**
   2. Press the **Submit** button

- **And There's So Much More to Learn!!!**