In-advance Data Analytics for Reducing Time to Discovery

Jialin Liu, Yin Lu, Yong Chen
Computer Science Department
Texas Tech University
Outline

- Introduction
- In-advance Data Analytics
  - Motivating Examples
  - Pattern Detection
  - In-memory Data Management
- Evaluation
- Conclusion
Introduction

- Big Data: Beyond the ability of commonly used software
- Difficult to discover knowledge hidden inside the data

Scientific Discovery Workflow

Data Generation → Data Movement → Data Analysis

Input Conditions → Insightful Information

IEEE BigData 2014
Scientific simulations tend to be data intensive

VPIC, a plasma physic simulation code, generates 1 trillion particles, with 26 bytes per particle, and total size of 36 TBs for one file, each time step (source: LBNL)

Collected data from instruments increases rapidly too

Scheduled to go live in 2020, the Large Synoptic Survey Telescope (LSST) will feature a 3.2-gigapixel camera capturing ultra-high-resolution images of the sky every 15 seconds, every night, for at least 10 years. Ultimately, the system will store more than 100 petabytes (about 20 million DVDs' worth) of data.
Motivation

➢ Reuse Sub Results

Compute Ensemble Mean of One variable

<table>
<thead>
<tr>
<th>Task 1</th>
<th>Task 2</th>
<th>Task 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>cdo ensmean in1 in2 in3 ofile1</td>
<td>cdo ensmean in3 in4 in5 ofile2</td>
<td>cdo ensmean in1 in2 in5 ofile3</td>
</tr>
</tbody>
</table>

➢ Predict Sub Results In-advance

Monthly Number of Clear/Cloudy Days

Operation 1: Compute Weights
nco -f nc -genbil, grid.txt -import_cmsaf CFCdm${day}.hdf we.nc

Operation 2: Remap and Selection
while [$day -le 20100831]
do
nco -f nc -gec, 75 -remap, grid.txt, we.nc -import_cmsaf CFCdm${day}.hdf CFCdm${day}.nc
day = 'expr $day + 1'
done

Operation 3: Add Values in All Files
nco enssum CFCdm201008??.nc cl.nc
Abstract of Workflow

How to reduce?
1. Partition the data into segments in task<sub>i</sub>.
2. Perform the predicted computation in-advance on each segment, Store the results as sub-results.
3. Reuse the sub-result in the task<sub>i+1</sub>.
Data Movement is Reduced by

- Case 1: Task 1 = Task 2; Data 1 ~ Data 2
- Case 2: Task 1 != Task 2; Data 1 ~ Data 2

Ideas: Predict and Reuse Sub-results

Challenges:

- How to predict the Task 2?
- How to detect the Data overlap?
- How to manage the sub-results?
Overview

Results

Pattern Detection

Result Generation

Reusing

Overlap Detection

I/O Optimization

Pattern Detection

History Ops

Pattern

Rec-op

(range, result, op)

In-memory K-V Store

File System

New Task

(\(OP, I/O\))

In-advance Analysis

Kernel

Overlap Detection

Optimized I/O

IEEE BigData 2014
Pattern Detection

- Pattern Definition
  - **Memorable** Pattern: Users’ analysis task is an ordered operation sequence. A latter operation in this sequence can not proceed without the intermediate results of the former operation in this sequence.
  - **Memoryless** Pattern: User’s future analytics only depend on the current operation, and are not related to the past operations.
Pattern Detection

- Directed Graph
- Succeeding Set
- Markov Chains
Pattern Detection

- Memorable Pattern
  - Succeeding(X) = \{operations come after X\}
    Succeeding(mean) = \{\text{min; ydayadd; yhourmul; enssum}\}

- Memoryless Pattern
  - \( P\{X^{(n+1)} = i \mid X^{(n)} = i_{n-1}, \ldots, X^{(0)} = i_0\} = P\{X^{(n+1)} = i \mid X^{(n)} = i_{n-1}\} \)
    Independent of time such that future operation only depends on the current
  - Transition probability matrix

\[
P = \begin{pmatrix}
P_{0,0} & P_{0,1} & \cdots & P_{0,m} \\
P_{1,0} & P_{1,1} & \cdots & P_{1,m} \\
\vdots & \vdots & \ddots & \vdots \\
P_{m,0} & P_{m,1} & \cdots & P_{m,m}
\end{pmatrix}
\]

\[
F = \begin{pmatrix}
F_{0,0} & F_{0,1} & \cdots & F_{0,m} \\
F_{1,0} & F_{1,1} & \cdots & F_{1,m} \\
\vdots & \vdots & \ddots & \vdots \\
F_{m,1} & \cdots & \cdots & F_{m,m}
\end{pmatrix}
\]

\[
P_{i,j} = \begin{cases}
\frac{F_{i,j}}{m} & \text{if } \sum_{i=1}^{m} F_{i,j} > 0 \\
\sum_{i=1}^{m} F_{i,j} & \text{if } \sum_{i=1}^{m} F_{i,j} = 0
\end{cases}
\]

IEEE BigData 2014
In-memory Data Management

Overlap Detection & Distributed In-memory Management

- Support Elastic Cloud Service in HPC
- Design Two-level Hash to Keep Track of the Sub-results

![Diagram of distributed in-memory database and MPI communication](image)

- Operation: (k,v) = (operation, data range) e.g., (max, [0 0 1k 2k 65536])
- Coarse hash

- Segment
- Finer hash: (k,v) = (operation, sub-range, sub-result) e.g., (min, [0 0 100 100 200], 0.3)

- Elastic: (IP: Port) 10.6.61.181:11211

---

IEEE BigData 2014
Evaluation

- Implementation
  - memcached 1.4.18, libmemcached 1.0.18, pnetcdf 1.4.1, CDO 1.6.
  - Initial connect with memcached, 1G per server
  - Dynamic request db server

- Preliminary Results (on TTU Hrothgar)
  - 180GB, 96 processes, 100 OST
Scientific applications become data intensive, the data movement and data management are bottleneck in scientific discovery.

In-advance Data Analytics Predicting users’ analysis operation, detecting the analysis pattern, and reusing the results for minimal data movement.

Future Work include increasing the fault tolerance and enabling the hybrid memory management
Thank you

For More Information: https://sites.google.com/site/jailinliu/