Concise representation of program execution as a job

Estimating derived data properties

Execution Engine

Data sizes: 10

internals are in Java)

map

out 1

MapReduce

100

Heavy use of programming languages for

Monitors task phases of

Job Profiling

map

c

raw data

/ Python

Business Analytics

Records information at the level of “task phases”

Wikipedia (compressed)

WC

Currently supports

Large space of tuning options

default FIFO scheduler

We currently use

map out 0

Supports concurrent job execution

Enable

the level

statistical

map

Wikipedia

Require no modifications to

WC

TeraGen

Estimated Surface

JO

TF

80

LG

= <

Job

TF

Profile fields separated into 4 categories

Whether output data should be compressed

Wikipedia

(40

Support unmodified

Program

Have zero overhead when profiling is turned off

Outputs the full description of given

job

One Reduce Wave

Two Map Waves

MapReduce Job Execution

✓ Job \( j \) \( = \langle \text{program} \ p, \ \text{data} \ d, \ \text{resource} \ r, \ \text{configuration} \ c \rangle \)

✓-supported concurrency

✓ Supports concurrently executing jobs

✓ Supports unmodified MapReduce

✓ Outputs the full profile description of MapReduce job’s behavior

Cost-based Optimization

✓ \( \text{perf} = F(p, d, r, c) \)

✓ \( c_{\text{opt}} = \arg \min_c F(p, d, r, c) \)

Tuning Challenges

✓ Heavy use of programming languages for MapReduce

✓ Programs (e.g., Java/Python)

✓ Large space of tuning options

✓ Number of map/reduce tasks

✓ Partitioning of map outputs to reduce tasks

✓ Memory allocation to task-level buffers

✓ Whether output data should be compressed

✓ Whether combine function should be used

MapReduce Job Profiling

✓ Concise representation of program execution as a job

✓ Records information at the level of “task phases”

✓ Profile fields separated into 4 categories

Job-level MapReduce configuration

Cluster sizing

Data layout tuning

Workflow

optimization

Workload

management

Dataflow: amount of data flowing

through task & task phases

Map output byte

- Number of map-side spills

- Spill phase time in the map task

- Number of records in buffer per spill

- Spill phase time in the map task

Dataflow: amount of data flowing

through task & task phases

Map output byte

- Number of map-side spills

- Spill phase time in the map task

- Number of records in buffer per spill

- Spill phase time in the map task

Cost: execution times at the level of

task & task phases

- Read phase time in the map task

- Split phase time in the map task

- Split phase time in the map task

Dataflow Statistics: statistical

information about the dataflow

- Map file’s selectivity (output/input)

- Map output compression rate

- Map output size (compressed)

- Size of records (keys and values)

- CPU cost for reading from local disk per byte

- CPU cost for executing Map fun per record

- CPU cost for uncompressing input per byte

Goals of Profiling

✓ Have zero overhead when profiling is turned off

✓ Require no modifications to Hadoop

✓ Support unmodified MapReduce programs written in Java or Hadoop Streaming/Pipes (Python/Ruby/C++)

Dynamic Instrumentation

✓ Monitors tasks phases of MapReduce job execution

✓ Event-condition-action rules are specified, leading to run-time instrumentation of Hadoop internals

✓ We currently use BTrace (Hadoop internals are in Java)

Generating Profiles by Measurement

✓ Profile Collection

✓ Task Execution

Generating Profiles by Estimation

✓ Given profile for \( j \) \( = \langle \phi, d_1, r_1, c_1 \rangle \)

✓ Estimate profile for \( j' \) \( = \langle \phi, d_2, r_2, c_2 \rangle \)

Cost Statistics:

- Relative to baseline

- Black-box Models

- White-box Models

Task Scheduler Simulator

✓ Simulates scheduling & execution of map and reduce tasks on a (hypothetical) cluster

✓ Supports concurrent job execution

✓ Currently supports Hadoop’s default FIFO scheduler

✓ Outputs the full description of given MapReduce job’s behavior

Usage of Virtual Job Profiles

- Estimating task execution times

- Estimating derived data properties

Cost-based Optimizer vs. Rule-based Manual Tuning

- Default Settings

- Rule-based Optimizer

- Cost-based Optimizer

Experimental Methodology

- 15-30 Amazon EC2 nodes, various instance types

- Cluster-level configurations based on rules of thumb

- Data sizes: 10-180 GB

Experimental Results

- Actual

- Predicted

- TS

- WC

- LG

- JO

- TF

- CO

- Amazon EC2

- MapReduce Programs

- CO

- Wikipedia

- TeraSort

- Business Analytics

- TeraGen

- MapReduce

- Wikipedia (compressed)

- TPCH

- Information Retrieval

- Wikipedia

- Speedup

- Running Time

- Overhead vs. Benefit of Profiling

- CO

- Wikipedia

- TeraSort

- Business Analytics

- TeraGen

- MapReduce

- Wikipedia (compressed)

- TPCH

- Information Retrieval

- Wikipedia