Introduction of BigDataBench 4.0

Wanling Gao
http://prof.ict.ac.cn

ICT, Chinese Academy of Sciences

ASPLOS 2018, Williamsburg, VA, USA
BigDataBench Tutorial Program

- 8:30-9:30 Wanling Gao
  - Introduction of BigDataBench 4.0

- 9:30-10:00 Chen Zheng
  - How to use BigDataBench 4.0

- 10:00-10:30 Coffee break

- 10:30-11:15 Chen Zheng
  - Big data and AI proxy benchmarks for simulation
First part

- Introduction of BigDataBench 4.0

- BigDataBench Benchmarking Methodology

- Simulation Benchmarks

- Characterization
Why Big Data and AI Benchmarking?

Measuring big data and AI systems, architectures quantitatively.
Challenge #1 Complexity

1) Portability cost
2) Reproducibility and interpretability of performance data

Complex

Fast changing

Expanding

Big data and AI workloads
Traditional Benchmarking Methodology

- Creating a new benchmark or proxy for every possible workload
- Case-by-case solution

Previous Benchmarking Methodology:
Custom implementation for every possible workload
What’s the Units of Computation?

So how to define a representative big data and AI benchmark suite?

*Big data and AI dwarf*: frequently-appearing units of computation in big data and AI workloads

--- a minimum set to represent maximum patterns
Challenge #2 Fairness

- No one-size-fits-all solution
  - Impact of data set
  - Impact of workloads
  - Impact of software stacks

- Many classes of big data and AI applications without comprehensive characterization
Challenge #3 Consistency

- Requirement difference between community
  - System: performance evaluation on large-scale system deployments
  - Architecture: heavily relies upon simulator-based research, needing shorter (simulation) runtime
  - AI researcher: runtime and model’s prediction precision

The benchmarks should be consistent across different communities for the co-design of software and hardware
Simulation for Big Data and AI

- Challenges
  - Simulators have limited supports on complex software stacks
    - For example: Hadoop modes
      - Standalone mode
      - Pseudo-distributed mode
      - Fully-distributed mode
  - Different modes have large behavior differences
- Long running time is unbearable
  - 1000+ times execution time than physical machine
What is BigDataBench?

- An open source big data benchmarking project
  - [http://prof.ict.ac.cn](http://prof.ict.ac.cn)
  - Search Google using “BigDataBench”
## BigDataBench 4.0 Overview

### BDGS (Big Data Generator Suite) for scalable data

<table>
<thead>
<tr>
<th>Wikipedia Entries</th>
<th>Amazon Movie Reviews</th>
<th>Google Web Graph</th>
</tr>
</thead>
<tbody>
<tr>
<td>Facebook Social Network</td>
<td>E-commerce Transaction</td>
<td>ProfSearch Resumes</td>
</tr>
<tr>
<td>ImageNet</td>
<td>CIFAR-10</td>
<td>LSUN</td>
</tr>
<tr>
<td>TED Talks</td>
<td>SoGou Data</td>
<td>MovieLens Dataset</td>
</tr>
<tr>
<td>MNIST</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### 13 Real-world Data Sets

- AI workloads
- Offline analytics
- Online service
- Streaming
- Graph analytics
- Data warehouse
- NoSQL workloads

### 47 Workloads with 7 types

- Micro benchmarks
- Component benchmarks
- Application benchmarks

### 16 Software Stacks

- Impala
- Flink
- GraphX
- Caffe
- Shark
- TensorFlow
- Hadoop
- RDMA
- MVAPICH
- DataMPI
- MPI
- NoSQL

---

**BigDataBench** | **ASPLOS 2018** | **ICT**

*INSTITUTE OF COMPUTING TECHNOLOGY, CHINESE ACADEMY OF SCIENCES*
What’s New in BigDataBench 4.0

- Dwarf-based benchmarking methodology
  - Micro, Component and Application Benchmarks

- Seven workload types
  - AI, Online service, Offline analytics, Graph analytics, Streaming, Data warehouse, NoSQL

- Dwarf-based simulation benchmarks
  - 100X runtime speedup, 90+% average accuracy
BigDataBench Evolution

- **BigDataBench 1.0**
  - Search engine
  - 6 workloads

- **BigDataBench 2.0**
  - Typical Internet service domains
  - An architectural perspective
  - 19 workloads & data generation tools

- **BigDataBench 3.0**
  - Multidisciplinary effort
  - 32 workloads: diverse implementations

- **BigDataBench 3.1**
  - 5 application domains: 14 data sets and 33 workloads
  - Same specifications: diverse implementations
  - Multi-tenancy version
  - BigDataBench subset and simulator version

- **BigDataBench 3.2**
  - New software stack: Flink, JStorm, GraphX, GraphLab
  - New workload type: Streaming, Graph processing
  - New dataset and workloads

- **BigDataBench 4.0**
  - Dwarf-based benchmarking methodology
  - Micro, component, application benchmark specification
  - 13 real-world data sets, 47 benchmarks, 7 workload types

- **BigDataBench Evolution**
  - New software stack: Flink, JStorm, GraphX, GraphLab
  - New workload type: Streaming, Graph processing
  - New dataset and workloads

- **Dwarf-based benchmarking methodology**
  - Micro, component, application benchmark specification
  - 13 real-world data sets, 47 benchmarks, 7 workload types
BigDataBench Users

- [http://prof.ict.ac.cn/BigDataBench/users/](http://prof.ict.ac.cn/BigDataBench/users/)

- Industry users
  - Accenture, BROADCOM, SAMSUMG, Huawei, IBM

- About 100 academia groups published papers using or citing BigDataBench
  - VLDB/SIGMOD, SC, FAST, ASPLOS, ISCA/Micro/HPCA, ICPP and etc.
## Why BigDataBench?

<table>
<thead>
<tr>
<th>Benchmarking Target</th>
<th>Methodology</th>
<th>Application domains</th>
<th>Workload types</th>
<th>Workloads</th>
<th>Scalable data sets abstracting from real data</th>
<th>Software Stacks</th>
</tr>
</thead>
<tbody>
<tr>
<td>BigDataBench 4.0</td>
<td>Dwarf-based</td>
<td>five</td>
<td>seven^1</td>
<td>forty-seven</td>
<td>13 real data sets; 6 scalable data sets</td>
<td>sixteen</td>
</tr>
<tr>
<td>BigDataBench 2.0</td>
<td>Popularity</td>
<td>three</td>
<td>three</td>
<td>nineteen</td>
<td>6 real data sets; 6 scalable data sets</td>
<td>ten</td>
</tr>
<tr>
<td>BigBench 2.0</td>
<td>Application model</td>
<td>one</td>
<td>five</td>
<td>Proposal</td>
<td>Proposal</td>
<td>Proposal</td>
</tr>
<tr>
<td>BigBench 1.0</td>
<td>Application model</td>
<td>one</td>
<td>one</td>
<td>ten</td>
<td>3 data generators</td>
<td>three</td>
</tr>
<tr>
<td>CloudSuite 3.0</td>
<td>Popularity</td>
<td>N/A</td>
<td>four</td>
<td>eight</td>
<td>3 data generators</td>
<td>three</td>
</tr>
<tr>
<td>HiBench 6.0</td>
<td>Popularity</td>
<td>N/A</td>
<td>six</td>
<td>nineteen</td>
<td>Random generate or with specific distribution</td>
<td>five</td>
</tr>
<tr>
<td>CALDA [13]</td>
<td>MapReduce system and parallel DBMSs</td>
<td>Popularity</td>
<td>N/A</td>
<td>one</td>
<td>five</td>
<td>N/A</td>
</tr>
<tr>
<td>YCSB [14]</td>
<td>Performance model</td>
<td>N/A</td>
<td>one</td>
<td>six</td>
<td>N/A</td>
<td>four</td>
</tr>
<tr>
<td>LinkBench [15]</td>
<td>Database systems</td>
<td>Application model</td>
<td>N/A</td>
<td>one</td>
<td>ten</td>
<td>one data generator</td>
</tr>
<tr>
<td>AMP Benchmarks [16]</td>
<td>Data analytic systems</td>
<td>Popularity</td>
<td>N/A</td>
<td>one</td>
<td>four</td>
<td>N/A</td>
</tr>
<tr>
<td>Fathom [17]</td>
<td>AI systems</td>
<td>Popularity</td>
<td>N/A</td>
<td>one</td>
<td>eight</td>
<td>N/A</td>
</tr>
</tbody>
</table>

The seven workload types are online service, offline analytics, graph analytics, artificial intelligence (AI), data warehouse, NoSQL, and streaming.
BigDataBench Publications

- Characterizing data analysis workloads in data centers. 2013 IEEE International Symposium on Workload Characterization (IISWC 2013)（Best paper award）
First part

- Introduction of BigDataBench 4.0
- *BigDataBench Benchmarking Methodology*
- Simulation Benchmarks
- Characterization
Dwarf-based Benchmarking Methodology

- A **scalable** dwarf-based benchmarking methodology
  - Combinations of dwarfs with different weights

![Dwarf-based Benchmarking Methodology: DAG-like combination](image)
Inspiration

**Successful Compute Abstractions**
- Relational algebra
  - 5 primitive operations
  - Select, Project, Product, Union, Difference
- Parallel computing
  - Computational & communication patterns
  - 13 dwarfs

**Successful Benchmarks**
- TPC-C
  - OLTP domain
  - Functions of abstraction
- HPCC
  - High performance computing
  - Seven basically tests
Dwarf Abstraction

- Big Data & AI Dwarf
  - Units of computation

- Dwarf Abstraction
  - Algorithmic analysis
  - Experimental analysis

**Big Data Analytics Workloads**

### Algorithmic Analysis
- Algorithm Decomposition
  - Units of Computation
  - DAG-like Combination

### Experimental Analysis
- Runtime Environment
- Operating System
- Architecture
- Runtime Tracing
- System Profiling
- Hardware Profiling

### Analysis & Verification
- Frequently-appearing units of computation
- Combination analysis
- Hotspot computations
- CPU Time breakdown
- CPU Cycle breakdown

### Hundreds of Dwarf Algorithms
- Abstraction of Frequently-appearing Units of Computation
- Minimum Set with Maximum Patterns
# Units of Computation

## Importance of eight classes units of computation

<table>
<thead>
<tr>
<th>Category</th>
<th>Application Domain</th>
<th>Workload</th>
<th>Unit of Computation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Graph Mining</td>
<td>Search Engine, Community Detection</td>
<td>PageRank, BFS, Connected component (CC)</td>
<td>Matrix, Graph, Sort</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Graph</td>
</tr>
<tr>
<td>Deminsion Reduction</td>
<td>Image Processing, Text Processing</td>
<td>Principal components analysis (PCA), LDA</td>
<td>Matrix, Basic Statistic, Sampling</td>
</tr>
<tr>
<td>Deep Learning</td>
<td>Image Recognition, Speech Recognition</td>
<td>Convolutional neural network (CNN), DBN</td>
<td>Matrix, Sampling, Transform</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Matrix, Sampling</td>
</tr>
<tr>
<td>Recommendation</td>
<td>Association Rules Mining, Electronic Commerce</td>
<td>Aporiorti, FP-Growth, Collaborative filtering (CF)</td>
<td>Basic Statistic, Graph, Set, Basic Statistic</td>
</tr>
<tr>
<td>Classification</td>
<td>Image Recognition, Speech Recognition, Text Recognition</td>
<td>Support vector machine (SVM), K-nearest neighbors (KNN), Naive bayes, Random forest, Decision tree (C4.5/CART/ID3)</td>
<td>Matrix, Sort, Basic Statistic, Graph, Basic Statistic</td>
</tr>
<tr>
<td>Clustering</td>
<td>Data Mining</td>
<td>K-means</td>
<td>Matrix, Sort</td>
</tr>
</tbody>
</table>
## Units of Computation (cont’)

- Importance of eight classes units of computation

<table>
<thead>
<tr>
<th>Feature Preprocess</th>
<th>Image Processing</th>
<th>Image segmentation (GrabCut)</th>
<th>Matrix, Graph</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Signal Processing</td>
<td>Scale-invariant feature transform (SIFT)</td>
<td>Matrix, Transform, Sampling, Sort, Basic Statistic</td>
</tr>
<tr>
<td></td>
<td>Text Processing</td>
<td>Image Transform</td>
<td>Matrix, Transform</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Term Frequency-inverse document frequency (TF-IDF)</td>
<td>Basic Statistic</td>
</tr>
<tr>
<td>Sequence Tagging</td>
<td>Bioinformatics</td>
<td>Hidden Markov Model (HMM)</td>
<td>Matrix</td>
</tr>
<tr>
<td></td>
<td>Language Processing</td>
<td>Conditional random fields (CRF)</td>
<td>Matrix, Sampling</td>
</tr>
<tr>
<td>Indexing</td>
<td>Search Engine</td>
<td>Inverted index, Forward index</td>
<td>Basic Statistic, Logic, Set, Sort</td>
</tr>
<tr>
<td>Encoding/Decoding</td>
<td>Multimedia Processing</td>
<td>MPEG-2</td>
<td>Matrix, Transform</td>
</tr>
<tr>
<td></td>
<td>Security</td>
<td>Encryption</td>
<td>Matrix, Logic</td>
</tr>
<tr>
<td></td>
<td>Cryptography</td>
<td>SimHash, MinHash</td>
<td>Set, Logic</td>
</tr>
<tr>
<td></td>
<td>Digital Signature</td>
<td>Locality-sensitive hashing (LSH)</td>
<td>Set, Logic</td>
</tr>
<tr>
<td>Data Warehouse</td>
<td>Business intelligence</td>
<td>Project, Filter, OrderBy, Union</td>
<td>Set, Sort</td>
</tr>
</tbody>
</table>
Big Data and AI Dwarfs

- Matrix computations
- Sampling computations
- Transform computations
- Graph computations
- Logic computations
- Set computations
- Basic statistic computations
- Sort computations

Figure 3.4: Gibbs sampling algorithm in two dimensions starting from an initial point and then completing three iterations.
One Combination Example

Feature extraction – SIFT Workload

Several dwarfs: \textit{transform} computations(FFT, IFFT), \textit{sampling} computations(downsampling), \textit{matrix} computations(matrix multiplication/subtraction), \textit{sort} computations(sort), \textit{basic statistic} computations(count)
Methodology Principle

Separating specification from implementation.
• Model relevant domains

State-of-the-art algorithms and technologies
• Implementation keep in pace with the improvement

Data impact
• Representative data sets considering typical types and sources
Five Steps

1. Investigate important application domain
2. Big data and AI dwarfs
3. Big data benchmark specification
4. Diverse implementation
5. Simulation Benchmark
Benchmarking Methodology

- **Specification**
  - Micro, component and application benchmark
Five Application Domains

DDBJ/EMBL/GenBank database Growth

Internet Service
Search engine, Social network, E-commerce

Multimedia

Bioinformatics

http://www.ddbj.nig.ac.jp/breakdown_stats/dbgrowth-e.html#dbgrowth-graph
BigDataBench 4.0 - Dataset

Un-structured

- Wikipedia Entries
- Amazon Movie Reviews
- MNIST
- SoGou Data
- MovieLens Dataset

Semi-structured

- E-commerce Transaction Data
- ProfSearch Person Resume

Structured

Table
- CIFAR-10
- ImageNet
- LSUN
- TED Talks

Graph
- Google Web Graph
- Facebook Social Graph

Multimedia

Un-structured
Big Data Generator Suite

BDGS Architecture
BDGS: Text Data Generator

- Modeling on topic and word level
- Words are drawn from distribution under particular topic
- Topics are selected from different distribution with parameters following a dirichlet distribution
# Micro Benchmarks

<table>
<thead>
<tr>
<th>Micro Benchmark</th>
<th>Involved Dwarf</th>
<th>Application Domain</th>
<th>Workload Type</th>
<th>Data Set</th>
<th>Software Stack</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sort</td>
<td>Sort</td>
<td>SE, SN, EC, MP, BI</td>
<td>Offline analytics</td>
<td>Wikipedia entries</td>
<td>Hadoop, Spark, Flink, MPI</td>
</tr>
<tr>
<td>Grep</td>
<td>Set</td>
<td></td>
<td>Offline analytics</td>
<td>Wikipedia entries</td>
<td>Hadoop, Spark, Flink, MPI</td>
</tr>
<tr>
<td>WordCount</td>
<td>Basic statistics</td>
<td></td>
<td>Offline analytics</td>
<td>Wikipedia entries</td>
<td>Hadoop, Spark, Flink, MPI</td>
</tr>
<tr>
<td>MD5</td>
<td>Logic</td>
<td></td>
<td>Offline analytics</td>
<td>Wikipedia entries</td>
<td>Hadoop, Spark, MPI</td>
</tr>
<tr>
<td>Connected Component</td>
<td>Graph</td>
<td>SN</td>
<td>Offline analytics</td>
<td>Facebook social network</td>
<td>Hadoop, Spark, Flink, GraphLab, MPI</td>
</tr>
<tr>
<td>RandSample</td>
<td>Sampling</td>
<td>SE, MP, BI</td>
<td>Offline analytics</td>
<td>Wikipedia entries</td>
<td>Hadoop, Spark, MPI</td>
</tr>
<tr>
<td>FFT</td>
<td>Transform</td>
<td>MP</td>
<td>Offline analytics</td>
<td>Two-dimensional matrix</td>
<td>Hadoop, Spark, MPI</td>
</tr>
<tr>
<td>Matrix Multiply</td>
<td>Matrix</td>
<td>SE, SN, EC, MP, BI</td>
<td>Offline analytics</td>
<td>Two-dimensional matrix</td>
<td>Hadoop, Spark, MPI</td>
</tr>
<tr>
<td>Read</td>
<td>Set</td>
<td>SE, SN, EC</td>
<td>NoSQL</td>
<td>ProfSearch resumes</td>
<td>HBase, MongoDB</td>
</tr>
<tr>
<td>Write</td>
<td>Set</td>
<td>SE, SN, EC</td>
<td>NoSQL</td>
<td>ProfSearch resumes</td>
<td>HBase, MongoDB</td>
</tr>
<tr>
<td>Scan</td>
<td>Set</td>
<td>SE, SN, EC</td>
<td>NoSQL</td>
<td>ProfSearch resumes</td>
<td>HBase, MongoDB</td>
</tr>
<tr>
<td>Convolution</td>
<td>Transform</td>
<td>SN, EC, MP, BI</td>
<td>AI</td>
<td>Cifar, ImageNet</td>
<td>TensorFlow, pyTorch, Caffe</td>
</tr>
<tr>
<td>Fully Connected</td>
<td>Matrix</td>
<td>SN, EC, MP, BI</td>
<td>AI</td>
<td>Cifar, ImageNet</td>
<td>TensorFlow, pyTorch, Caffe</td>
</tr>
<tr>
<td>Relu</td>
<td>Logic</td>
<td>SN, EC, MP, BI</td>
<td>AI</td>
<td>Cifar, ImageNet</td>
<td>TensorFlow, pyTorch, Caffe</td>
</tr>
<tr>
<td>Sigmoid</td>
<td>Matrix</td>
<td>SN, EC, MP, BI</td>
<td>AI</td>
<td>Cifar, ImageNet</td>
<td>TensorFlow, pyTorch, Caffe</td>
</tr>
<tr>
<td>Tanh</td>
<td>Matrix</td>
<td>SN, EC, MP, BI</td>
<td>AI</td>
<td>Cifar, ImageNet</td>
<td>TensorFlow, pyTorch, Caffe</td>
</tr>
<tr>
<td>MaxPooling</td>
<td>Sampling</td>
<td>SN, EC, MP, BI</td>
<td>AI</td>
<td>Cifar, ImageNet</td>
<td>TensorFlow, pyTorch, Caffe</td>
</tr>
<tr>
<td>AvgPooling</td>
<td>Sampling</td>
<td>SN, EC, MP, BI</td>
<td>AI</td>
<td>Cifar, ImageNet</td>
<td>TensorFlow, pyTorch, Caffe</td>
</tr>
<tr>
<td>BatchNorm [37]</td>
<td>Basic Statistics</td>
<td>SN, EC, MP, BI</td>
<td>AI</td>
<td>Cifar, ImageNet</td>
<td>TensorFlow, pyTorch, Caffe</td>
</tr>
<tr>
<td>Dropout [38]</td>
<td>Sampling</td>
<td>SN, EC, MP, BI</td>
<td>AI</td>
<td>Cifar, ImageNet</td>
<td>TensorFlow, pyTorch, Caffe</td>
</tr>
</tbody>
</table>
# Component Benchmarks

<table>
<thead>
<tr>
<th>Component Benchmark</th>
<th>Involved Dwarf</th>
<th>Application Domain</th>
<th>Workload Type</th>
<th>Data Set</th>
<th>Software Stack</th>
</tr>
</thead>
<tbody>
<tr>
<td>Xapian Server</td>
<td>Get, Put, Post</td>
<td>SE</td>
<td>Online service</td>
<td>Wikipedia entries</td>
<td>Xapian</td>
</tr>
<tr>
<td>PageRank</td>
<td>Matrix, Sort, Basic statistics, Graph</td>
<td>SE</td>
<td>Graph analytics</td>
<td>Google web graph</td>
<td>Hadoop, Spark, Flink, GraphLab, MPI</td>
</tr>
<tr>
<td>Index</td>
<td>Logic, Sort, Basic statistics, Set</td>
<td>SE</td>
<td>Offline analytics</td>
<td>Wikipedia entries</td>
<td>Hadoop, Spark</td>
</tr>
<tr>
<td>Rolling top words</td>
<td>Sort, Basic statistics</td>
<td>SN</td>
<td>Streaming</td>
<td>Random generate</td>
<td>Spark streaming, JStorm</td>
</tr>
<tr>
<td>Kmeans</td>
<td>Matrix, Sort, Basic statistics</td>
<td>SE, SN, EC, MP, BI</td>
<td>Offline analytics</td>
<td>Facebook social network</td>
<td>Hadoop, Spark, Flink, MPI</td>
</tr>
<tr>
<td>Collaborative Filtering</td>
<td>Graph, Matrix</td>
<td>EC</td>
<td>Offline analytics</td>
<td>Amazon movie review</td>
<td>Hadoop, Spark</td>
</tr>
<tr>
<td>Naive Bayes</td>
<td>Basic statistics, Sort</td>
<td>SE, SN, EC</td>
<td>Offline analytics</td>
<td>Amazon movie review</td>
<td>Hadoop, Spark, Flink, MPI</td>
</tr>
<tr>
<td>SIFT</td>
<td>Matrix, Sampling, Transform, Sort</td>
<td>MP</td>
<td>Offline analytics</td>
<td>ImageNet</td>
<td>Hadoop, Spark, MPI</td>
</tr>
<tr>
<td>LDA</td>
<td>Matrix, Graph, Sampling</td>
<td>SE</td>
<td>Offline analytics</td>
<td>Wikipedia entries</td>
<td>Hadoop, Spark, MPI</td>
</tr>
<tr>
<td>OrderBy</td>
<td>Set, Sort</td>
<td>EC</td>
<td>Data warehouse</td>
<td>E-commerce transaction</td>
<td>Hive, Spark-SQL, Impala</td>
</tr>
<tr>
<td>Aggregation</td>
<td>Set, Basic statistics</td>
<td>EC</td>
<td>Data warehouse</td>
<td>E-commerce transaction</td>
<td>Hive, Spark-SQL, Impala</td>
</tr>
<tr>
<td>Project, Filter</td>
<td>Set</td>
<td>EC</td>
<td>Data warehouse</td>
<td>E-commerce transaction</td>
<td>Hive, Spark-SQL, Impala</td>
</tr>
<tr>
<td>Select, Union</td>
<td>Set</td>
<td>EC</td>
<td>Data warehouse</td>
<td>E-commerce transaction</td>
<td>Hive, Spark-SQL, Impala</td>
</tr>
<tr>
<td>Alexnet</td>
<td>Matrix, Transform, Sampling, Logic, Basic statistics</td>
<td>SN, MP, BI</td>
<td>AI</td>
<td>Cifar, ImageNet</td>
<td>TensorFlow, pyTorch</td>
</tr>
<tr>
<td>Googlenet</td>
<td>Matrix, Transform, Sampling, Logic, Basic statistics</td>
<td>SN, MP, BI</td>
<td>AI</td>
<td>Cifar, ImageNet</td>
<td>TensorFlow, pyTorch</td>
</tr>
<tr>
<td>Resnet</td>
<td>Matrix, Transform, Sampling, Logic, Basic statistics</td>
<td>SN, MP, BI</td>
<td>AI</td>
<td>Cifar, ImageNet</td>
<td>TensorFlow, pyTorch</td>
</tr>
<tr>
<td>Inception Resnet V2</td>
<td>SN, MP, BI</td>
<td>AI</td>
<td>Cifar, ImageNet</td>
<td>TensorFlow, pyTorch</td>
<td></td>
</tr>
<tr>
<td>VGG16</td>
<td>SN, MP, BI</td>
<td>AI</td>
<td>Cifar, ImageNet</td>
<td>TensorFlow, pyTorch</td>
<td></td>
</tr>
<tr>
<td>DCGAN</td>
<td>SN, MP, BI</td>
<td>AI</td>
<td>LSUN</td>
<td>TensorFlow, pyTorch</td>
<td></td>
</tr>
<tr>
<td>WGAN</td>
<td>SN, MP, BI</td>
<td>AI</td>
<td>LSUN</td>
<td>TensorFlow, pyTorch</td>
<td></td>
</tr>
<tr>
<td>GAN</td>
<td>SN, MP, BI</td>
<td>AI</td>
<td>LSUN</td>
<td>TensorFlow, pyTorch</td>
<td></td>
</tr>
<tr>
<td>Seq2Seq</td>
<td>SE, EC, BI</td>
<td>AI</td>
<td>TED Talks</td>
<td>TensorFlow, pyTorch</td>
<td></td>
</tr>
</tbody>
</table>
First part

- Introduction of BigDataBench 4.0
- BigDataBench Benchmarking Methodology
- Simulation Benchmarks
- Characterization
Dwarf-based Simulation Methodology

- DAG-like combinations of dwarfs
  - Different weights
  - Computation logic
Simulation Benchmark for Big Data

- Simulation benchmarks for Hadoop workloads
  - 100X runtime speedup
  - 90+% data accuracy

- OpenMP & Pthread Implementations
  - Provide a unified memory management module
    - Mimic JVM garbage collection (GC) process
Memory Management Module
Dwarf Components for Big Data

- Data generation tools
- Dwarf implementations (OpenMP & Pthreads)
Simulation Benchmarks for AI

- Dwarf implementations (OpenMP & Pthreads)

1) Convolution
2) Fully connected
3) Relu
4) Dropout
5) Sigmoid
6) Tanh
7) Max
8) Avg pooling
9) Batch Norm

Neural Network
First part

- Introduction of BigDataBench 4.0
- BigDataBench Benchmarking Methodology
- Simulation Benchmarks
- Characterization
Workload Characterization

Hardware
- 3-node Hadoop cluster
  - Network: 1 Gb Ethernet network
  - Processor: Intel Xeon E5-2620 v3 (Haswell)

<table>
<thead>
<tr>
<th>Hardware Configurations</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU Type</td>
<td>Intel CPU Core</td>
</tr>
<tr>
<td>Intel @Xeon E5-2620 V3</td>
<td>12 <a href="mailto:cores@2.40G">cores@2.40G</a></td>
</tr>
<tr>
<td>L1 DCache</td>
<td></td>
</tr>
<tr>
<td>12 x 32 KB</td>
<td>12 x 32 KB</td>
</tr>
<tr>
<td>L1 ICache</td>
<td></td>
</tr>
<tr>
<td>12 x 32 KB</td>
<td>12 x 256 KB</td>
</tr>
<tr>
<td>L2 Cache</td>
<td>15MB</td>
</tr>
<tr>
<td>L3 Cache</td>
<td></td>
</tr>
<tr>
<td>Memory</td>
<td>64GB,DDR4</td>
</tr>
<tr>
<td>Disk</td>
<td>SATA@7200RPM</td>
</tr>
<tr>
<td>Ethernet</td>
<td>1Gb</td>
</tr>
<tr>
<td>Hyper-Threading</td>
<td>Disabled</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Software Configurations</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating System</td>
<td>CentOS 7.2</td>
</tr>
<tr>
<td>Linux Kernel</td>
<td>4.1.13</td>
</tr>
<tr>
<td>JDK Version</td>
<td>1.8.0_65</td>
</tr>
<tr>
<td>Hadoop Version</td>
<td>2.7.1</td>
</tr>
<tr>
<td>Hive Version</td>
<td>0.9.0</td>
</tr>
<tr>
<td>HBase Version</td>
<td>1.0.1</td>
</tr>
<tr>
<td>Spark Version</td>
<td>1.5.2</td>
</tr>
<tr>
<td>Tensorflow Version</td>
<td>1.0</td>
</tr>
</tbody>
</table>

Software
- Software version
  - CentOS 7.2, Kernel 4.1.13
  - JDK version: 1.8.0_65
  - Hadoop version: 2.7.1

- Compared benchmarks
  - SPEC CPU2006
  - HPCC 1.4.0
  - PARSEC 2.0

Benchmark
- Seven workload types
Execution Performance

- ILP and MLP
  - AI: ILP slightly lower than SPECCPU, MLP similar with HPCC
  - Big data has lower ILP and MLP than AI for almost all types, except Hive based data warehouse type
Pipeline Efficiency (Level 1)

- AI reflect similar pipeline behaviors with the traditional benchmarks
  - retiring (35% v.s. 39.8%), bad speculation (6.3% v.s. 6.1%), frontend bound (both about 9%), and backend bound (49.7% v.s. 45.1%)
- Big data and AI have a small fraction of bad speculation

![Graph showing pipeline efficiency](image-url)
About Detailed Characterization

- BPOE workshop tomorrow will give detailed characterization results

- Look forward to your participation!
Download

- http://prof.ict.ac.cn/download

- Packing & Testing now!

- Release soon (April 1st, 2018)
Conclusion

- BigDataBench 4.0
  - An open source dwarf-based big data and AI benchmark suite

- Website: http://prof.ict.ac.cn

- Technical Reports: