BigDataBench Tutorial, Part 2: Benchmarking methodology, models and metrics

Jianfeng Zhan

ICT, Chinese Academy of Sciences & BenchCouncil

HPCA 2019, Washington D.C., USA
Outline

- The challenges and motivation of characterizing modern workloads
- What is BigDataBench?
- Tutorial introduction
- Benchmarking methodology, model and metrics
- The summary of BigDataBench 5.0
Basic Methodology

Big Data and AI Workloads

Algorithmic Analysis
- Pipeline of units of computation
- Data input and intermediate data

Profiling Analysis
- Computation graph
- Run time breakdown

Data Motifs
- Frequently-appearing Units of Computation
- Data Inputs (type, source, size, pattern)
Algorithms with a broad spectrum

- Internet services
- Data mining/Machine learning
- Natural language processing/Computer vision (Recognition Sciences)
- Bioinformatics (Medical Sciences)
What is Data motif?

- Data Motif: abstractions of time-consuming units of computation
- classes of units of computation
- The impacts of data type, source, size, pattern

Our observations: eight Data Motifs

- Matrix
- Sampling
- Transform
- Graph
- Logic
- Set
- Statistics
- Sort
Expression power of eight data motifs

- Using the combination of data motifs to represent a wide variety of big data and AI workloads

**Big Data and AI Motif**
- Matrix
- Sampling
- Transform
- Graph
- Logic
- Set
- Statistics
- Sort

**Combinations with different weights**

**Diverse big data and AI workloads**
- Coverage of fundamental units of computation
- Provide the methodology of choosing typical workloads
- Reduce workload redundancy
Matrix computation

- Operations on one/multiple rectangular arrays of numbers or other objects
  - Vector-vector
  - Vector-matrix
  - Matrix-matrix

\[(i_1, i_2) \cdot \begin{bmatrix}
  w_{11} & w_{12} & w_{13} \\
  w_{21} & w_{22} & w_{23}
\end{bmatrix} = (o_1, o_2, o_3)\]
Sampling

- The selection of a subset of original data.
  - Random sampling
  - Importance sampling
  - Acceptance sampling
  - Monte Carlo sampling
Transform computation

- Equation from its original domain into another domain.
  - Fourier transform
  - Laplace transform
Graph computation

- Nodes represent entities and edges represent dependencies.
  - Community Detection
  - PageRank
Logic computation

- Bit manipulation
  - AND
  - OR
  - XOR

![Logic computation diagram with examples of MD5 hash for different inputs: M, My Name, My Name is John.](image-url)
Set computation

- Operations on one/multiple collection of distinct objects
  - Set theory
    - Union
    - Intersection
    - Complement
  - Similarity analysis
Sort

- Sorting algorithm that puts elements of a list in a certain order
  - Top-K Sort
  - Memory sort
  - External sort
  - Sort algorithms
    - QuickSort
    - BubbleSort
Basic statistic computation

- Data models and statistics
  - Probability distribution
  - Count statistics
  - Time-series analysis
AI Workload --- AlexNet

Units of Computation:

1) Convolution: 36.91%
   ---- Conv2d

2) Sampling: 13.45%
   ---- Max Pooling
   ---- Dropout

3) Matrix Multiply: 48.87%
   ---- Fully Connected

4) Basic Statics: 0.76%
   ---- Normalization
Feature Extraction --- SIFT

SIFT: Units of Computation

1) Builds Gaussian pyramid: 13.16%
   - Matrix Multiplication
   - Transform
   - DownSample

2) Builds DoG pyramid: 4.17%
   - Matrix Subtraction
   - Matrix Inversion

3) Finds keypoints: 26.01%
   - Sort
   - Matrix Inversion

4) Compute scale, orientation & descriptors: 53.11%
   - Statistic

5) Sort: 0.53%
   - Sort
Data motifs’ Differences from Kernels

- behaviors are affected by the sizes, patterns, types, and sources of different data inputs

- reflect not only *computation* patterns, *memory access* patterns, but also *disk and network I/O* patterns
Domain-specific Hardware and Software Co-design

- Ad-hoc solution
  - Case by case

- Structure solution
  - Tailoring the system and architecture to characteristics of data motifs
    - New architecture/accelerator design
    - Data motif-based libraries
    - Bottleneck identification and optimization
BigDataBench benchmarking models

- BigDataBench provides three benchmark models for evaluating hardware, software system, and algorithms, respectively.
BigDataBench intact Model Division

- This model is for hardware benchmarking.

- The users should run the implementation on their hardware directly without modification.

- The only allowed tuning includes hardware, OS and compiler settings.
BigDataBench constrained Model Division

- This model is for software system benchmarking.
- The division specifies the model to be used and restricts the values of hyper parameters, e.g. batch size and learning rate.
- The users can implement the algorithms on their software platforms or frameworks by themselves.
BigDataBench free Model Division

- This model is for algorithm benchmarking.
- The users are specified with using the same data set
- with the emphasis being on advancing the state-of-the-art of algorithms.
Metrics (1)

- For the BigDataBench intact Model Division, the metrics include the wall clock time and energy efficiency to run benchmarks.
Metrics (2)

- For the BigDataBench constrained model division, the metrics include the wall clock time and energy efficiency to run benchmarks. In addition, the values of hyper parameters should be reported for audition.
For the BigDataBench free model division, the metrics include the accuracy, and the wall clock time and energy efficiency to run benchmarks.