ビッグデータ･AI時代のクラスタ型スーパコンピュータ: 東工大TSUBAME3.0から産総研ABCI (AI Bridging Cloud Infrastructure)へ

松岡　聡

教授: 東工大 GSIC & 特定フェロー: 産総研AIRC

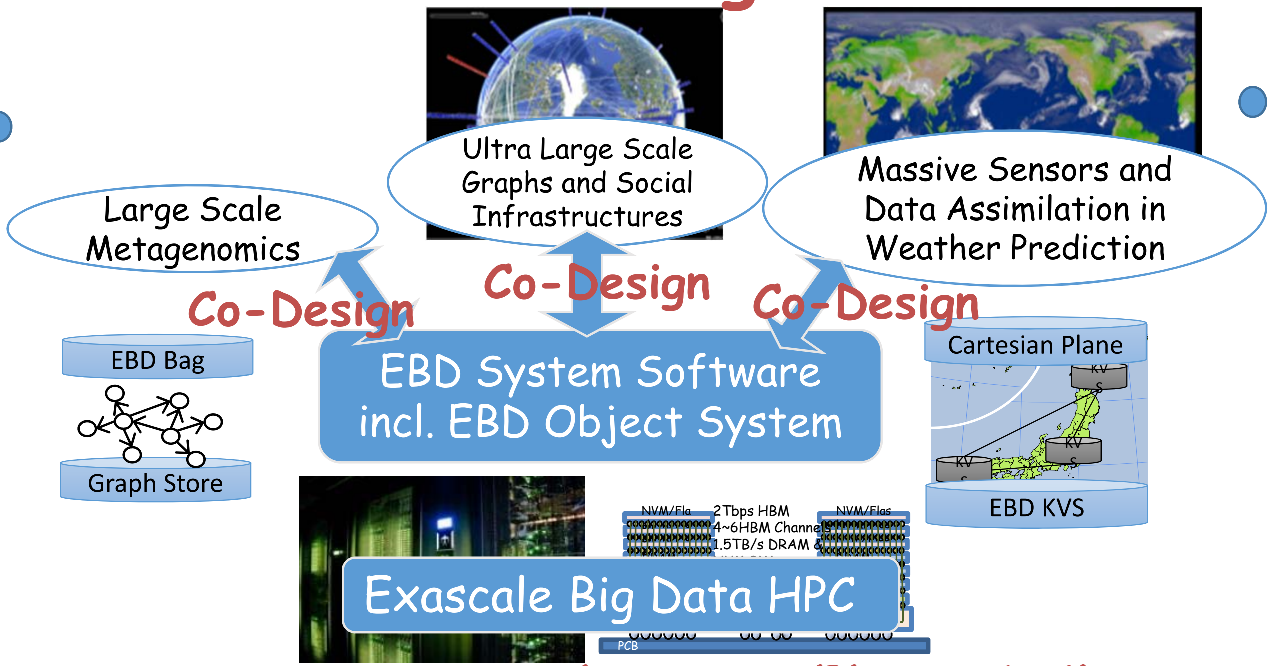
2016年12月16日

PCクラスタコンソシウム講演@秋葉原

JST-CREST “Extreme Big Data” Project (2013-2018)

Future Non-Silo Extreme Big Data Scientific Apps

Given a top-class supercomputer, how fast can we accelerate next generation big data c.f. Clouds?

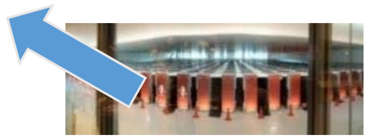


Convergent Architecture (Phases 1~4)
Large Capacity NVM, High-Bisection NW

Issues regading Architectural, algorithmic, and system software evolution?

Use of GPUs?

Cloud IDC
Very low BW & Efficiency
Highly available, resilient



Supercomputers
Compute&Batch-Oriented
More fragile

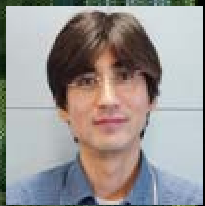
Extreme Big Data (EBD) Team

Co-Design EHPC and EDB Apps

- **Satoshi Matsuoka (PI), Toshio Endo, Hitoshi Sato (Tokyo Tech.) (EBD Software System)**



- **Osamu Tatebe (Univ. Tsukuba) (EBD-I/O)**



- **Michihiro Koibuchi (NII) (EBD Network)**

- **Yutaka Akiyama, Ken Kurokawa (Tokyo Tech) (EBD App1 Genome)**


- **Takemasa Miyoshi (Riken AICS) (EBD App2 Weather, data assim.)**

- **Toyotaro Suzumura (IBM Watson / Columbia U)(EBD App3 Social Simulation)**
- **(now merged into Matsuoka Team)**

EBD Recent Awards & Accolades

- IEEE Sidney Fernbach Award (2014, Matsuoka)
- Rakuten Technology Award (2014, Matsuoka)
- HPCWire 2015 Readers Choice Awards Outstanding Leadership in HPC (2015/11)
  - Satoshi Matsuoka, co-award with Prof. Jack Dongarra@U-Tennessee
- World No.1 in Graph500 Benchmark on K computer(2016/11)
  - 4 consecutive wins: 2015/06, 2015/11, 2016/06, 2016/11
- IPSJ Computer Science Research Award for Young Scientists (2016/10)
  - Keita Iwabuchi, “Towards a Distributed Large-Scale Dynamic Graph Data Store”, IPSJ SIG Technical Report Vol. 2015-HPC-153, 2015/03
- 2015年度情報処理学会長尾真記念特別賞(2016/06)
  - 鯉渕 道紘「ラックスケールコンピュータ・ネットワークの設計に関する先駆的な研究」


日経テクノロジーonline

ニュース

「京」がGraph500で3期連続世界第1位

2016/07/14 09:32 1/1ページ

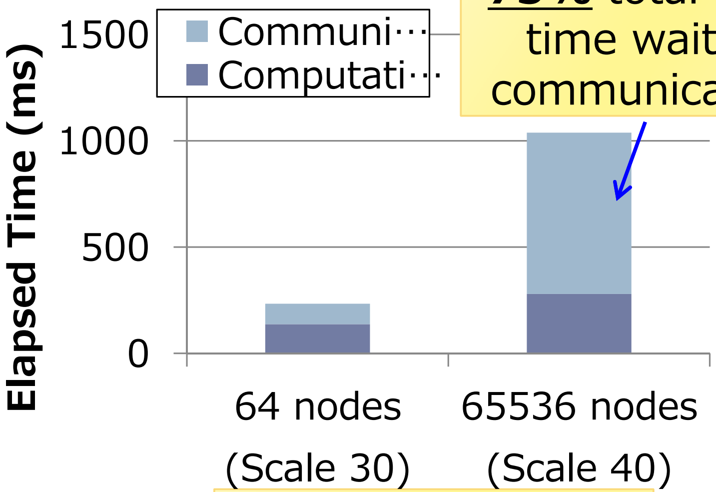
九州大学と東京工業大学、理化学研究所、スペインのバルセロナ・スーパーコンピューティング・センター、富士通による国際共同研究グループは2016年7月13日、ビッグデータ処理で重要となる大規模グラフ解析に関するスーパーコンピューターの国際的な性能ランキング「Graph500」（2016年6月20日公開）において、スーパーコンピューター「京（けい）」が第1位を獲得したと発表した。2015年11月に続き3期連続（通算4期）の第1位となる。

Open Source Release of EBD System Software (install on T3/Amazon/ABCI)

- mrCUDA
  - rCUDA extension enabling remote-to-local GPU migration
  - https://github.com/EBD-CREST/mrCUDA
  - GPU 3.0
  - Co-Funded by NVIDIA
- CBB
  - I/O Burst Buffer for Inter Cloud Environment
  - https://github.com/EBD-CREST/cbb
  - Apache License 2.0
  - Co-funded by Amazon
- ScaleGraph Python
  - Python Extension for ScaleGraph X10-based Distributed Graph Library
  - https://github.com/EBD-CREST/scalegraphpython
  - Eclipse Public License v1.0
- GPUSort
  - GPU-based Large-scale Sort
  - https://github.com/EBD-CREST/gpusort
  - MIT License
- Others in development…

The Graph500 – 2015~2016 – 4 Consecutive world #1

K Computer #1 Tokyo Tech[EBD CREST] Univ. Kyushu [Fujisawa Graph CREST], Riken AICS, Fujitsu



73% total exec time wait in communication

88,000 nodes,
660,000 CPU Cores
1.3 Petabyte mem
20GB/s Tofu NW

K computer

≫

Effective x13 performance c.f. Linpack

List	Rank	GTEPS	Implementat
November 2013	4	5524.12	Top-down o
June 2014	1	17977.05	Efficient hybrid
November 2014	2		Efficient hybrid
June, Nov 2015 June Nov 2016	**1**	**38621.4**	**Hybrid + Node Compression**

*Problem size is weak scaling "Brain-class" graph

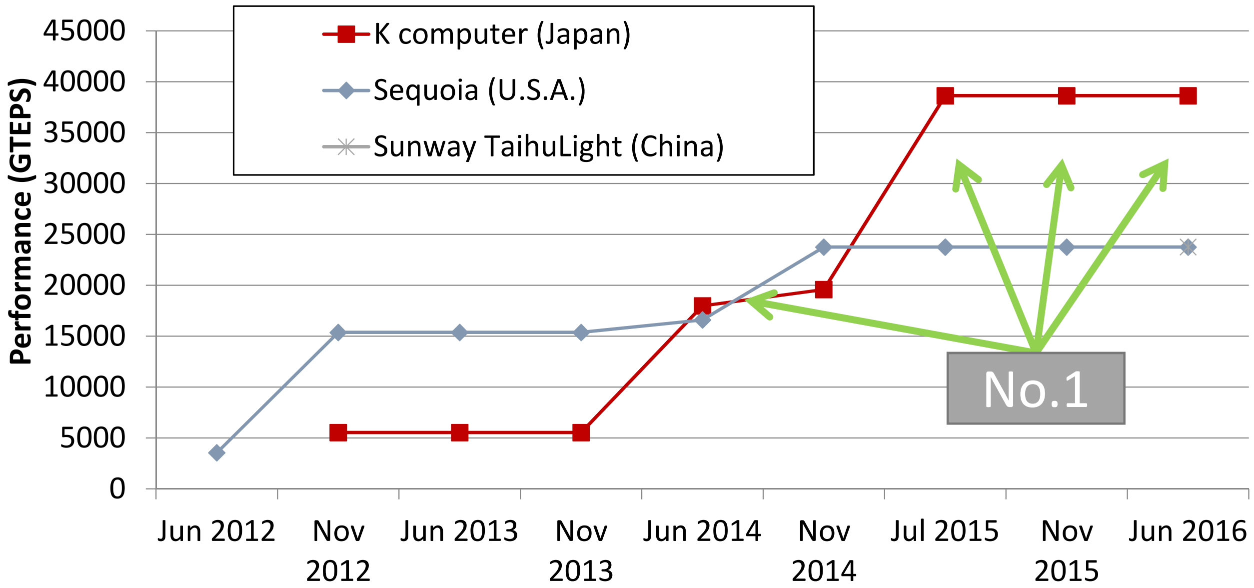
LLNL-IBM Sequoia
1.6 million CPUs
1.6 Petabyte mem

TaihuLight
10 million CPUs
1.3 Petabyte mem

IBM

K-computer No.1 on Graph500: 4th Consecutive Time

- What is Graph500 Benchmark?
  - Supercomputer benchmark for data intensive applications.
  - Rank supercomputers by the performance of **Breadth-First Search** for very huge graph data.



This is achieved by a combination of high machine performance and **our software optimization**.

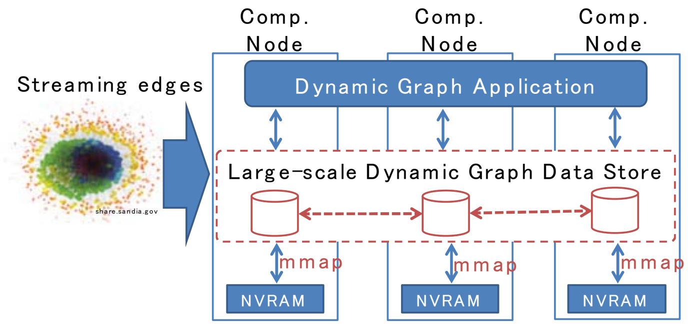
- Efficient Sparse Matrix Representation with Bitmap
- Vertex Reordering for Bitmap Optimization
- Optimizing Inter-Node Communications
- Load Balancing

etc.

- Koji Ueno, Toyotaro Suzumura, Naoya Maruyama, Katsuki Fujisawa, and Satoshi Matsuoka, "**Efficient Breadth-First Search on Massively Parallel and Distributed Memory Machines**", in proceedings of 2016 IEEE International Conference on Big Data (IEEE BigData 2016), Washington D.C., Dec. 5-8, 2016 (to appear)

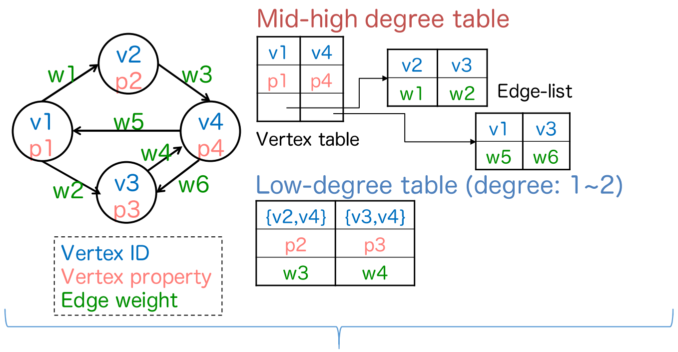
Towards a Distributed Large-Scale Dynamic Graph Data Store

Goal: to develop the data store for large-scale dynamic graph analysis on supercomputers



Node Level Dynamic Graph Data Store

Follows an adjacency-list format and leverages an open address hashing to construct its tables



Extend for multi-processes using an async MPI communication framework

Dynamic Graph Construction (on-memory)

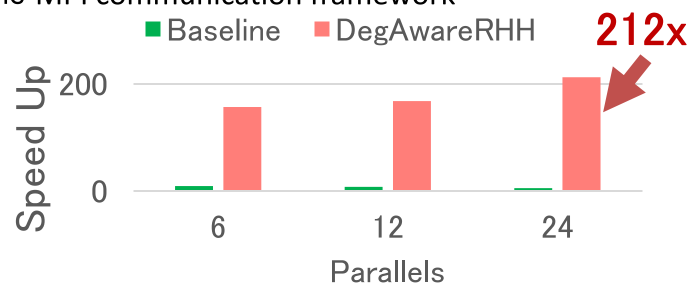
Against STINGER (single-node)

STINGER

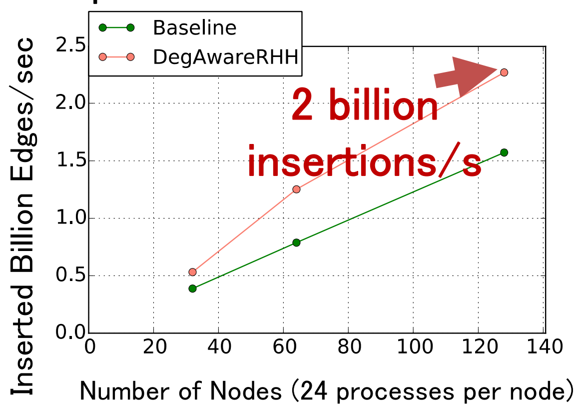
- A state-of-the-art dynamic graph processing framework developed at Georgia Tech

Baseline model

- A naïve implementation using *Boost* library (C++) and the MPI communication framework



Multi-node Experiment

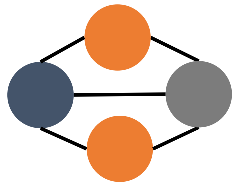


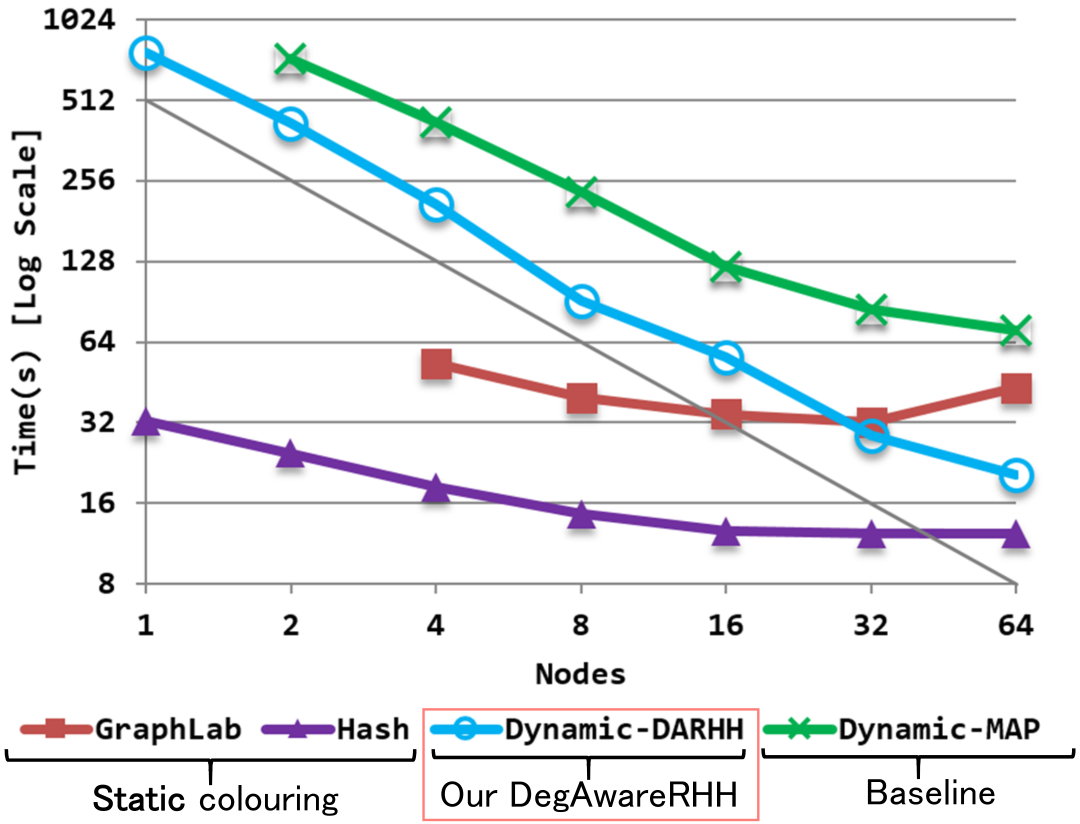
K. Iwabuchi, S. Sallinen, R. Pearce, B. V. Essen, M. Gokhale, and S. Matsuoka, Towards a distributed large-scale dynamic graph data store. In 2016 IEEE Interna- tional Parallel and Distributed Processing Symposium Workshops (IPDPSW)

Large-scale Graph Colouring (vertex coloring)

SC'16

- Color each vertices with the minimal #colours so that **no** two adjacent vertices have the same colour
- Compare our dynamic graph colouring algorithm on DegAwareRHH against:
  1. two static algorithms including GraphLab
  2. an another graph store implementation with same dynamic algorithm (Dynamic-MAP)

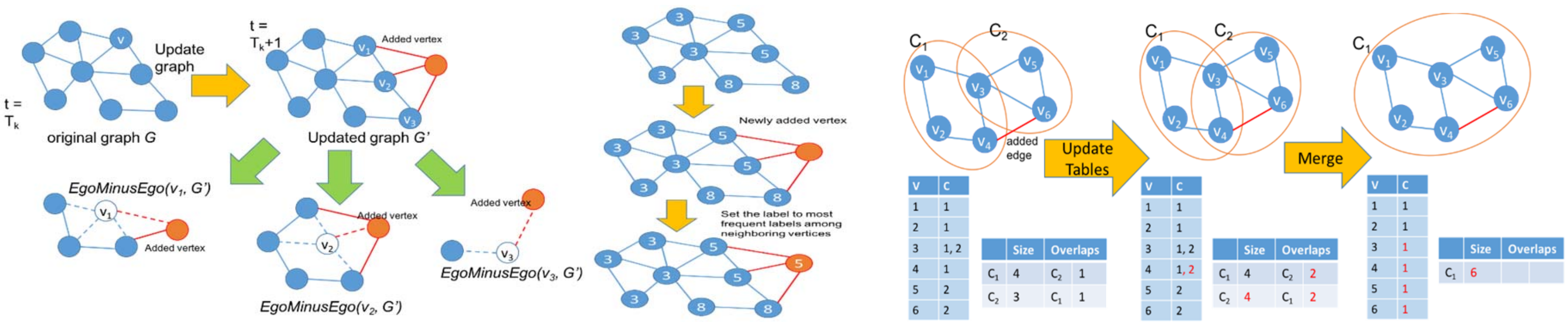


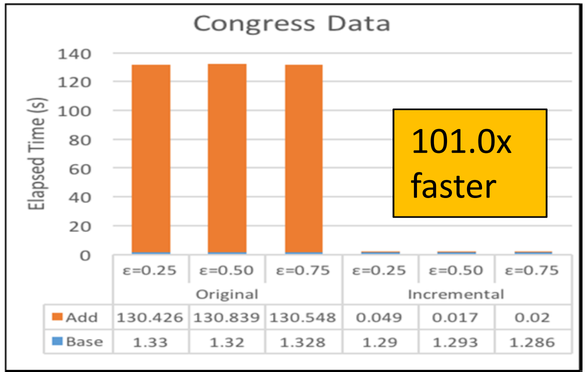


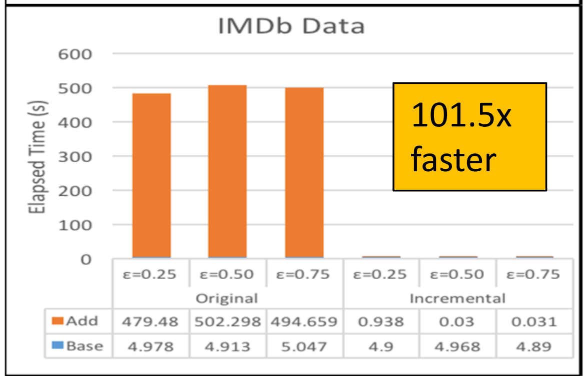
Scott Sallinen, Keita Iwabuchi, Roger Pearce, Maya Gokhale, Matei Ripeanu, "Graph Coloring as a Challenge Problem for Dynamic Graph Processing on Distributed Systems", SC' 16

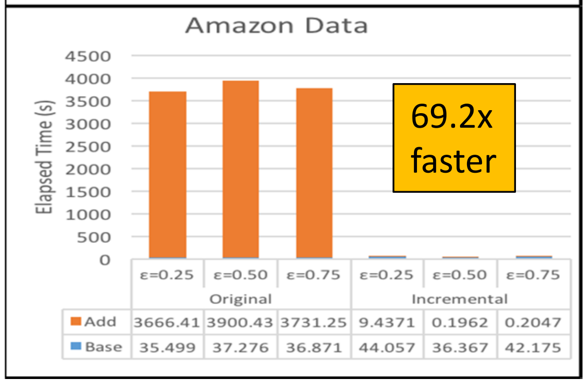
Incremental Graph Community Detection

- Background
  - Community detection for large-scale **time-evolving and dynamic graphs** has been one of important research problems in graph computing.
  - It is time-wasting to compute communities entire graphs every time from scratch.
- Proposal
  - **An incremental community detection algorithm** based on core procedures in a state-of-the-art community detection algorithm named DEMON.
    - <u>Ego Minus Ego</u>, <u>Label Propagation</u> and <u>Merge</u>









Hiroki Kanezashi and Toyotaro Suzumura, An Incremental Local-First Community Detection Method for Dynamic Graphs, Third International Workshop on High Performance Big Graph Data Management, Analysis, and Mining (BigGraphs 2016), to appear

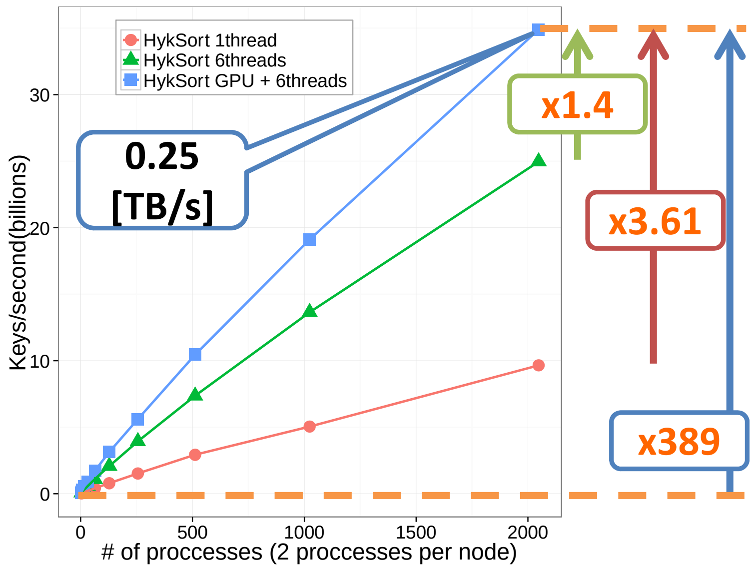
GPU-based Distributed Sorting [Shamoto, IEEE BigData 2014, IEEE Trans. Big Data 2015]

EBD Algorithm Kernels

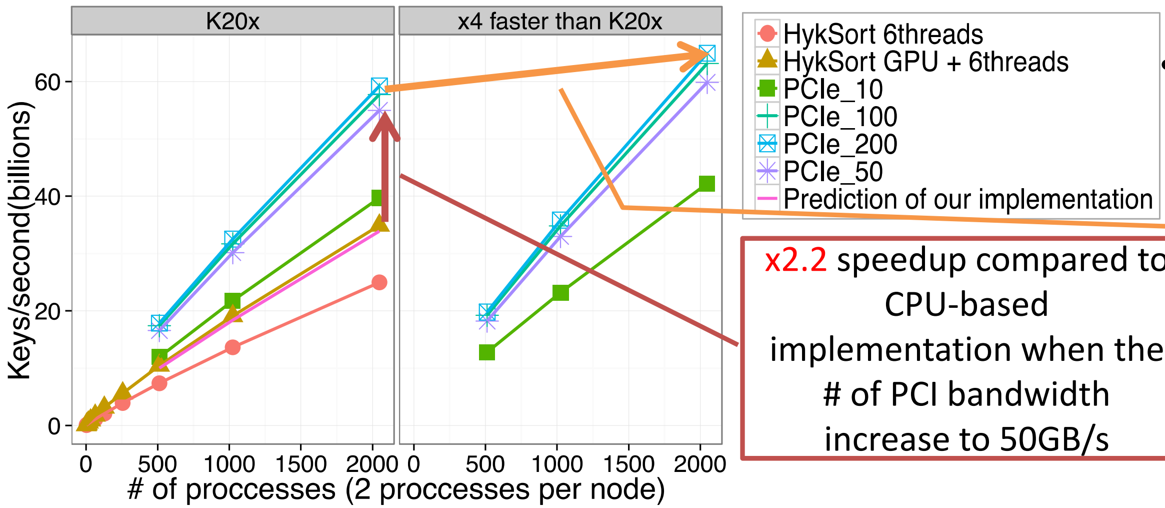
- Sorting: Kernel algorithm for various EBD processing
- Fast sorting methods
  - Distributed Sorting: Sorting for distributed system
    - Splitter-based parallel sort
    - Radix sort
    - Merge sort
  - Sorting on heterogeneous architectures
    - Many sorting algorithms are accelerated by many cores and high memory bandwidth.
- Sorting for large-scale heterogeneous systems remains unclear
- We develop and evaluate bandwidth and latency reducing GPU-based HykSort on TSUBAME2.5 via latency hiding
  - Now preparing to release the sorting library

GPU implementation of splitter-based sorting (HykSort)

- Weak scaling performance (Grand Challenge on TSUBAME2.5)
  - 1 ~ 1024 nodes (2 ~ 2048 GPUs)
  - 2 processes per node
  - Each node has 2GB 64bit integer
- C.f. Yahoo/Hadoop Terasort: 0.02[TB/s]
  - Including I/O



Performance prediction



- PCIe_#: #GB/s bandwidth of interconnect between CPU and GPU

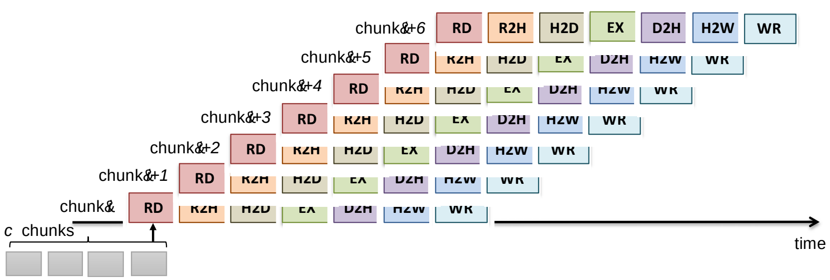
x2.2 speedup compared to CPU-based implementation when the # of PCI bandwidth increase to 50GB/s

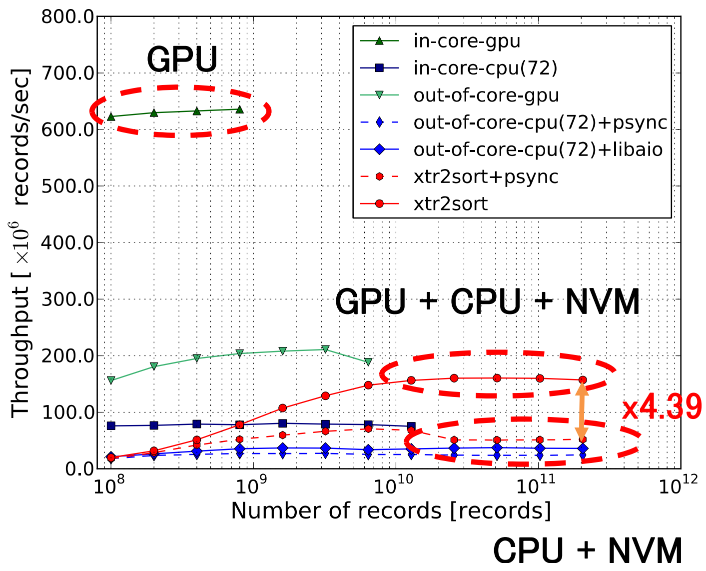
8.8% reduction of overall runtime when the accelerators work 4 times faster than K20x

Xtr2sort: Out-of-core Sorting Acceleration using GPU and Flash NVM [IEEE BigData2016]

How to combine deepening memory layers for future HPC/Big Data workloads, targeting Post Moore Era?

- Sample-sort-based Out-of-core Sorting Approach for Deep Memory Hierarchy Systems w/ GPU and Flash NVM
  - I/O chunking to fit device memory capacity of GPU
  - Pipeline-based Latency hiding to overlap data transfers between NVM, CPU, and GPU using asynchronous data transfers, e.g., cudaMemCpyAsync(), libaio



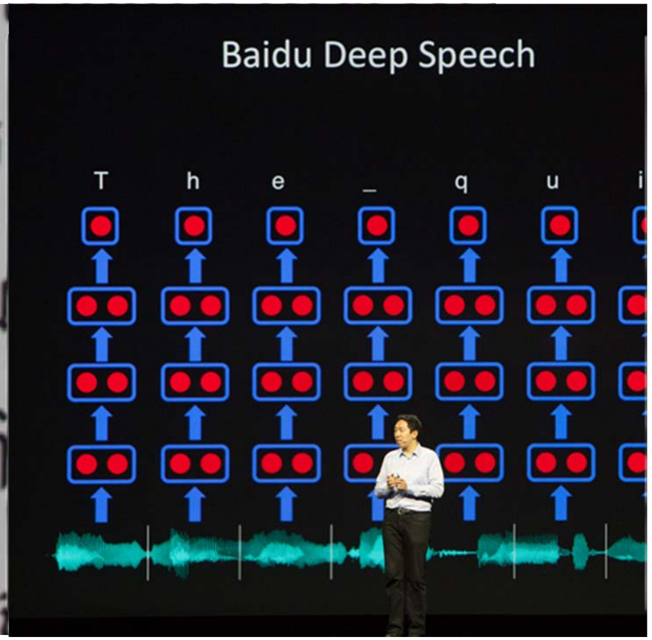


Deep Learning Shock

NVIDA GTC2015 March 2015

Elon Musk@TESLA

Jeff Dean@Google



Andrew Ng@Baidu

Deep Learning Shock

GTC2015 March 2015

Elon Musk@T A   Jeff Dean@Go   Andrew Ng@Ba

Deep Learning Rules!
HPC important basis

Deep Learning Rules!
HPC important basis

Deep Learning Rules!
HPC important basis

Deep Learning Shock
NVIDIA GTC2015 March 2015

Comment from Anonymous TSUBAME user

We have thousands of GPUs
and I can use them at will;
But I know nothing about DL
Despite being vendor talks
it seems to have wide-ranging apps,
making it mainstream HPC
So how are we going to even
survive?

2011 ACM Gordon Bell Prize Winner & First Author

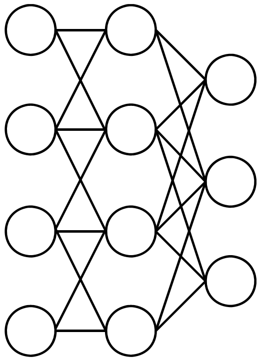
Just Putting in a few Deep Learning Libraries on Custom-Made Supercomputers will not Infrastructural Proliferation…

So how? Well we already know how to make things go viral on the Internet!

Background
Deep Learning (DL)

- A machine learning technique using “Deep” Neural Network
  - DL is achieving state-of-the-art in large machine learning area
  - Training DNN with huge dataset requires large scale computation
    - eg. 15-layer CNN training takes 8.2 days on 16 nodes (48 GPUs) of TSUBAME2.5
    - Researchers have to train DNN for several times to optimize DNN structure and hyper-parameters by hand

Imput(Image)
(Reference: http://image-net.org/)



Neural Network

Barn swallow = **0.95**
Police dog = 0.03
Water beetle = 0.01
⋮

Output(Classes)

Estimated Compute Resource Requirements for Deep Learning [Source: Preferred Network Japan Inc.]

To complete the learning phase in one day

P:Peta
E:Exa
F:Flops

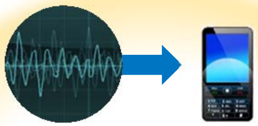
Image/Video Recognition

10P（Image）～ 10E（Video） Flops
学習データ：1億枚の画像 10000クラス分類
数千ノードで6ヶ月 [Google 2015]

Bio / Healthcare

100P ～ 1E Flops
一人あたりゲノム解析で約10M個のSNPs
100万人で100PFlops、1億人で1EFlops

Image Recognition



10P～ Flops
1万人の5000時間分の音声データ
人工的に生成された10万時間の
音声データを基に学習 [Baidu 2015]

Auto Driving

1E～100E Flops
自動運転車１台あたり1日 1TB
10台～1000台, 100日分の走行データの学習

Robots / Drones

1E～100E Flops
1台あたり年間1TB
100万台～1億台から得られた
データで学習する場合

機械学習、深層学習は学習データが大きいほど高精度になる
現在は人が生み出したデータが対象だが、今後は機械が生み出すデータが対象となる

各種推定値は1GBの学習データに対して1日で学習するためには
1TFlops必要だとして計算

10PF	100PF	1EF	10EF	100EF
2015	2020	2025	2030	

Performance Modeling of a Large Scale Asynchronous Deep Learning System under Realistic SGD Settings – *Detailed Performance Modeling and Optimization of Scalable DNN*

Yosuke Oyama[1], Akihiro Nomura[1], Ikuro Sato[2], Hiroki Nishimura[3], Yukimasa Tamatsu[3], and Satoshi Matsuoka[1]
[1]Tokyo Institute of Technology [2]DENSO IT LABORATORY, INC. [3]DENSO CORPORATION

[To Appear IEEE Big Data 2016]

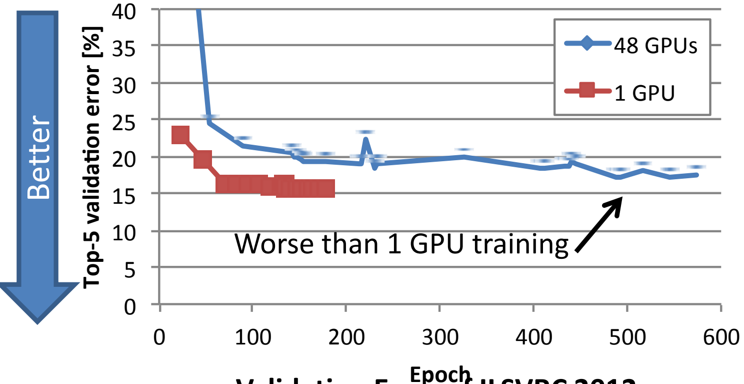
DENSO

Background

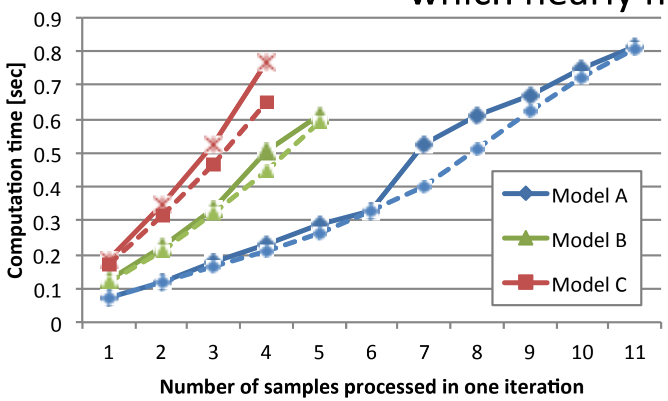
- Deep Convolutional Neural Networks (DCNNs) have achieved stage-of-the-art performance in various machine learning tasks such as image recognition
- Asynchronous Stochastic Gradient Descent (SGD) method has been proposed to accelerate DNN training
  - It may cause unrealistic training settings and degrade recognition accuracy on large scale systems, due to large non-trivial mini-batch size

Proposal and Evaluation

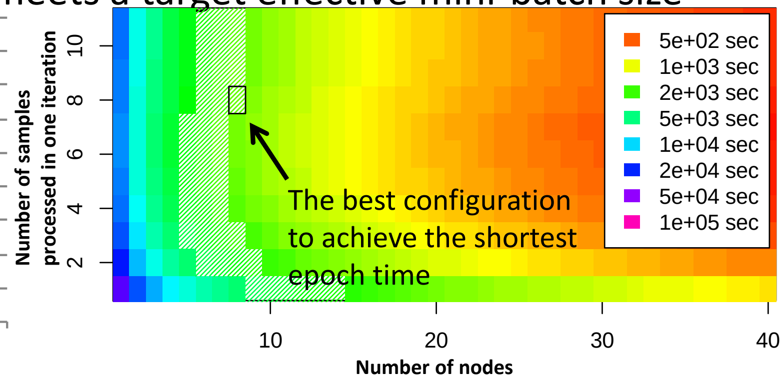
- We propose a empirical performance model for an ASGD training system on GPU supercomputers, which predicts CNN computation time and time to sweep entire dataset
  - Considering "effective mini-batch size", time-averaged mini-batch size as a criterion for training quality
- Our model achieves 8% prediction error for these metrics in average on a given platform, and steadily choose the fastest configuration on two different supercomputers which nearly meets a target effective mini-batch size



Validation Error of ILSVRC 2012 Classification Task on Two Platforms:
Trained 11 layer CNN with ASGD method



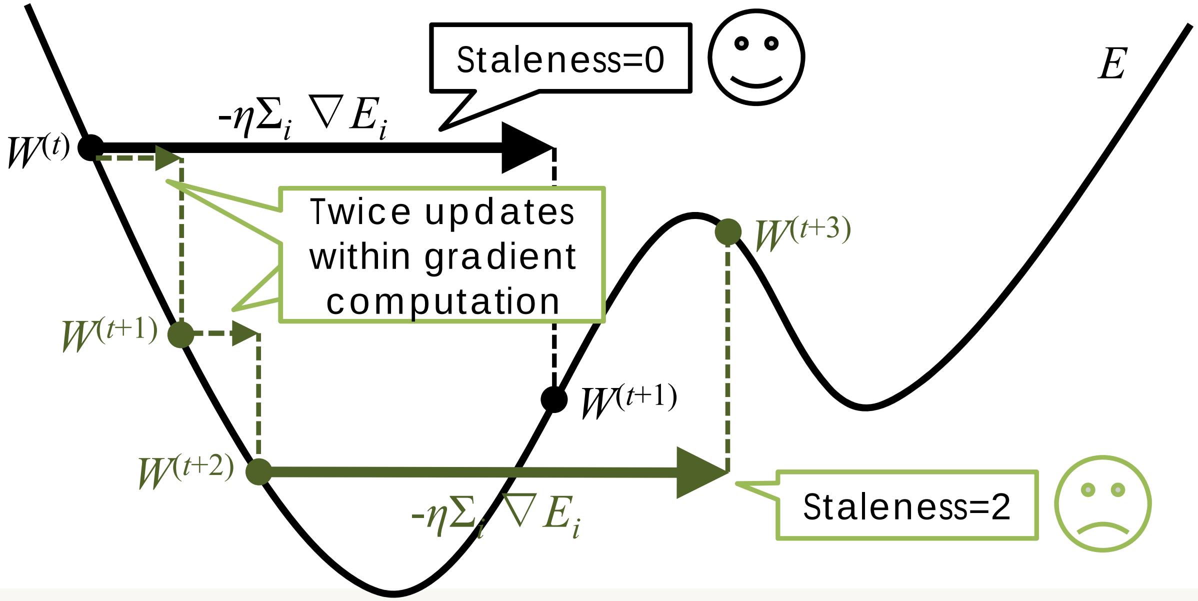
Measured Time (Solid) and Predicted Time (Dashed) of CNN Computation of Three 15-17 Layer Models



Predicted Epoch Time of ILSVRC 2012 Classification Task:
Shaded area indicate the effective mini-batch size is in 138±25%

Background
Mini-batch Size and Staleness

- **Staleness**
  - # of updates done within one gradient computation
- Existing researches showed that the error is increased by larger mini-batch size and staleness
  - **There was no way of knowing these statistics in advance**



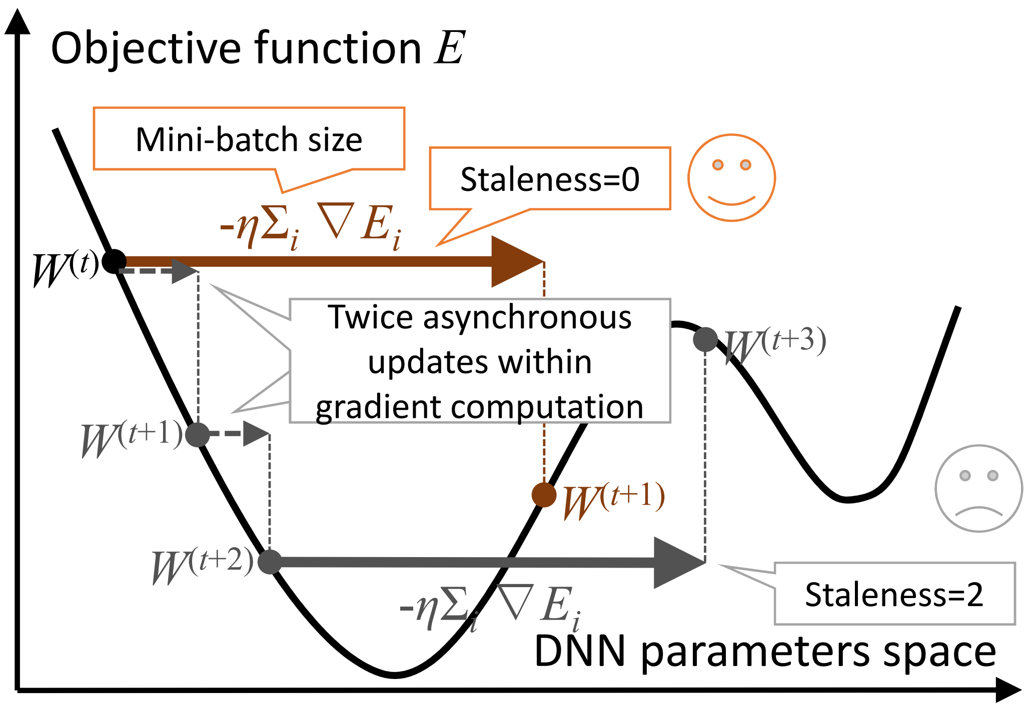
Approach and Contribution

- **Approach**: Proposing a performance model for an ASGD deep learning system SPRINT which considers probability distribution of mini-batch size and staleness
  - Takes CNN structure and machine specifications as input
  - Predicts time to sweep entire dataset (epoch time) and the distribution of the statistics
- **Contribution**
  - Our model predicts epoch time, average mini-batch size and staleness with 5%, 9%, 19% error in average respectively on several supercomputers
  - Our model steadily choose the fastest machine configuration that nearly meets a target mini-batch size

Predicting Statistics of Asynchronous SGD Parameters for a Large-Scale Distributed Deep Learning System on GPU Supercomputers

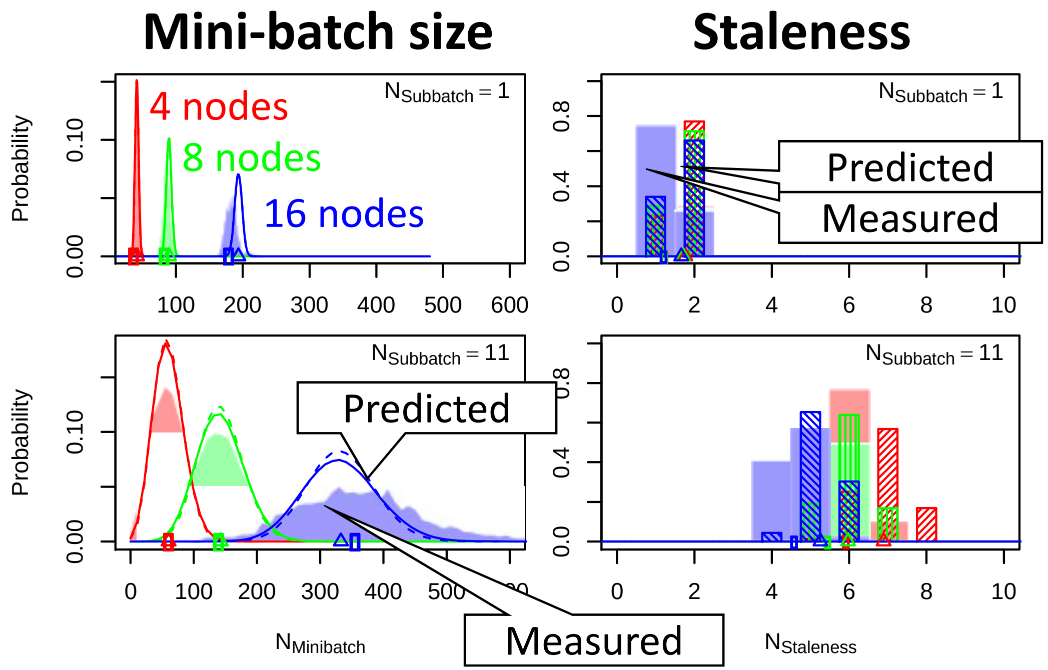
Background

- In large-scale Asynchronous Stochastic Gradient Descent (ASGD), mini-batch size and gradient staleness tend to be large and unpredictable, which increase the error of trained DNN



Proposal

- We propose a empirical performance model for an ASGD deep learning system SPRINT which considers probability distribution of mini-batch size and staleness



($N_{Subbatch}$: # of samples per one GPU iteration)

- Yosuke Oyama, Akihiro Nomura, Ikuro Sato, Hiroki Nishimura, Yukimasa Tamatsu, and Satoshi Matsuoka, "**Predicting Statistics of Asynchronous SGD Parameters for a Large-Scale Distributed Deep Learning System on GPU Supercomputers**", in proceedings of 2016 IEEE International Conference on Big Data (IEEE BigData 2016), Washington D.C., Dec. 5-8, 2016 (to appear)

Performance Prediction of Future HW for CNN

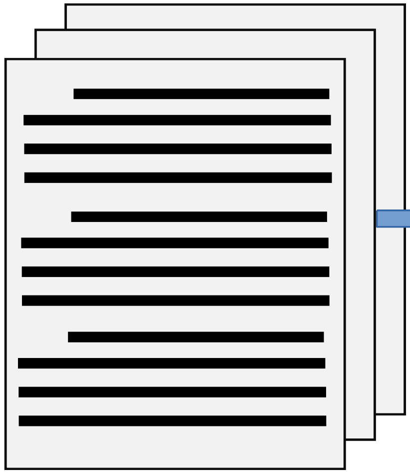
- 以下の2点を用いた想定で最適なパラメータを予測
  - FP16: 半精度浮動小数点数を用いた計算・データ保持性能の向上
  - EDR IB: 4xEDR InfiniBand (12.5GB/s)を用いたノード間通信性能の向上

→ Not only # of nodes, but also fast interconnect is important for scalability

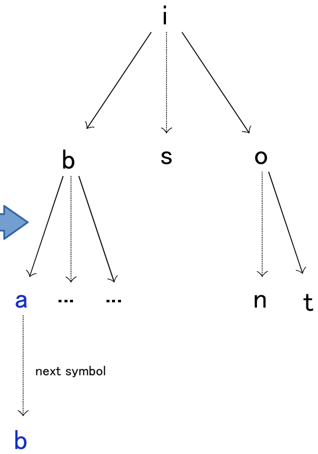
TSUBAME-KFC/DLでのILSVRC2012データセットの学習における最適なパラメータの予測 (平均ミニバッチサイズ138±25%)

	N_Node	N_Subbatch	Epoch時間	平均ミニバッチサイズ
(現在のHW)	8	8	1779	165.1
FP16	7	22	1462	170.1
EDR IB	12	11	1245	166.6
FP16 + EDR IB	8	15	1128	171.5

Algorithms and Tools for Vector Space Models of Big Data Computational Linguistics

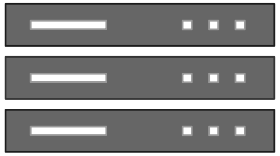


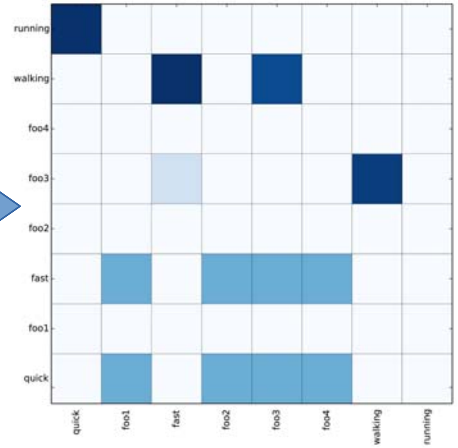
Corpus: large collection of texts



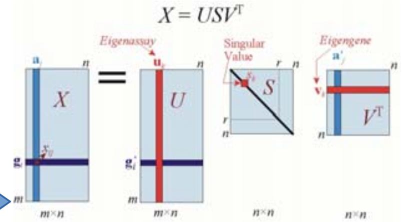
Efficient and compact intra-node data structures based on ternary trees

Distributed implementation with MPI library





Co-occurrence matrix in Compressed Sparse Row format stored to parallel storage in hdf5



Integration with SLEPC high performance distributed sparse linear algebra library for dimensionality reduction

Various machine learning algorithms

The whole pipeline orchestrated from Python

http://vsm.blackbird.pw

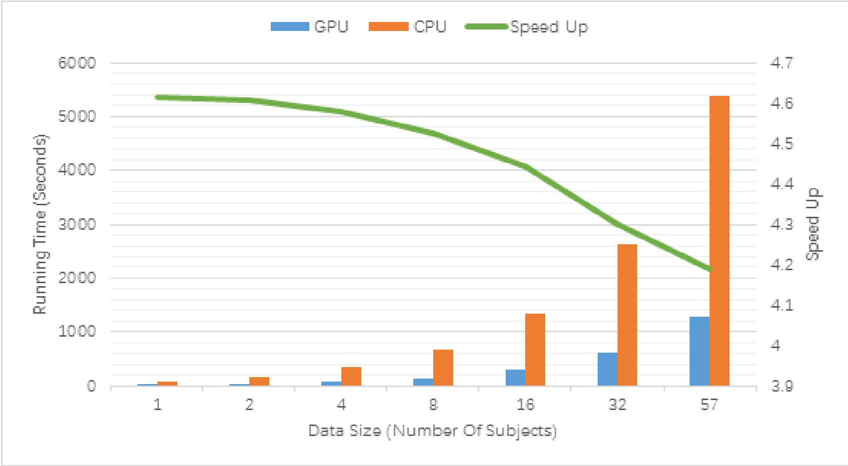
Publications:

- A. Drozd, A. Gladkova, and S. Matsuoka, “Word embeddings, analogies, and machine learning: beyond king - man + woman = queen,” accepted for COLING 2016.
- A. Gladkova, A. Drozd, and S. Matsuoka, “Analogy-based detection of morphological and semantic relations with word embeddings: what works and what doesn’t.,” in Proceedings of the NAACL-HLT SRW, San Diego, California, June 12-17, 2016, 2016, pp. 47–54.
- A. Gladkova and A. Drozd, “Intrinsic evaluations of word embeddngs: what can we do better?,” in Proceedings of The 1st Workshop on Evaluating Vector Space Representations for NLP, Berlin, Germany, 2016, pp. 36–42.
- A. Drozd, A. Gladkova, and S. Matsuoka, “Discovering Aspectual Classes of Russian Verbs in Untagged Large Corpora”, Proceedings of 2015 IEEE International Conference on Data Science and Data Intensive Systems (DSDIS), 2015, pp. 61–68.
- A. Drozd, A. Gladkova, and S. Matsuoka, “Python, Performance, and Natural Language Processing,” in Proceedings of the 5th Workshop on Python for High-Performance and Scientific Computing, New York, NY, USA, 2015, p. 1:1–1:10.

GPU-Based Fast Signal Processing for Large Amounts of Snore Sound Data

- Background

Snore sound (SnS) data carry very important information for diagnosis and evaluation of Primary Snoring and Obstructive Sleep Apnea (OSA). With the increasing number of collected SnS data from subjects, how to handle such large amount of data is a big challenge. In this study, we utilize the Graphics Processing Unit (GPU) to process a large amount of SnS data collected from two hospitals in China and Germany to accelerate the features extraction of biomedical signal.



Results of GPU and CPU based systems for processing SnS data

- Acoustic features of SnS data

we extract **11** acoustic features from a large amount of SnS data, which can be visualized to help doctors and specialists to diagnose, research, and remedy the diseases efficiently.

Snore sound data information

Subjects	Total Time (hours)	Data Size (GB)	Data format	Sampling Rate
57 (China + Germany)	187.75	31.10	WAV	16 kHz, Mono

- Result

We set 1 CPU (with Python2.7, numpy 1.10.4 and scipy 0.17 packages) for processing 1 subject's data as our baseline. Result show that the GPU based system is almost 4.6× faster than the CPU implementation. However, the speed-up decreases when increasing the data size. We think that this result should be caused by the fact that, the transmission of data is not hidden by other computations, as will be a real-world application.

* Jian Guo, Kun Qian, Huijie Xu, Christoph Janott, Bjorn Schuller, Satoshi Matsuoka, "GPU-Based Fast Signal Processing for Large Amounts of Snore Sound Data", *In proceedings of 5th IEEE Global Conference on Consumer Electronics (GCCE 2016)*, October 11-14, 2016.