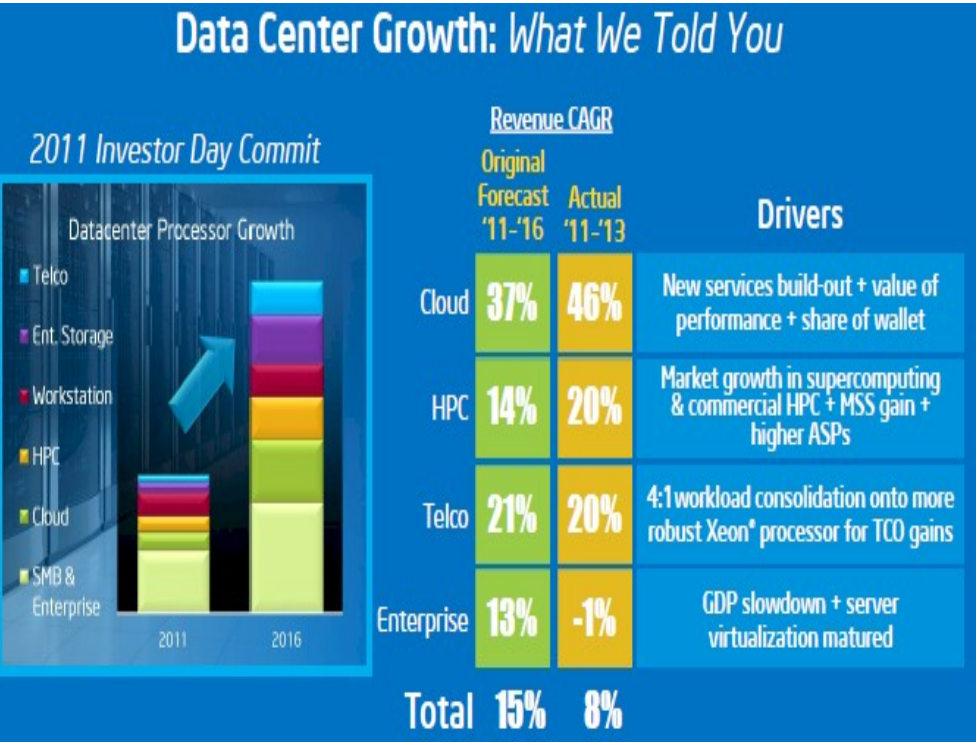
中科院计算所
INSTITUTE OF COMPUTING TECHNOLOGY,CAS

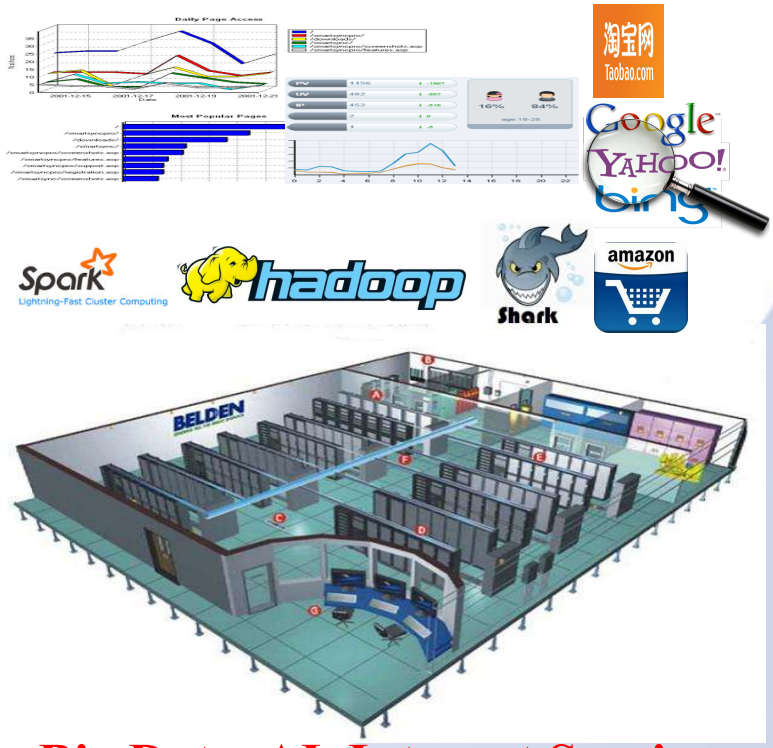
BOPS, Not FLOPS! A New Metric, Measuring Tool, and Roofline Performance Model For Datacenter Computing

Chen Zheng
ICT，CAS

Data Center Computing (DC)



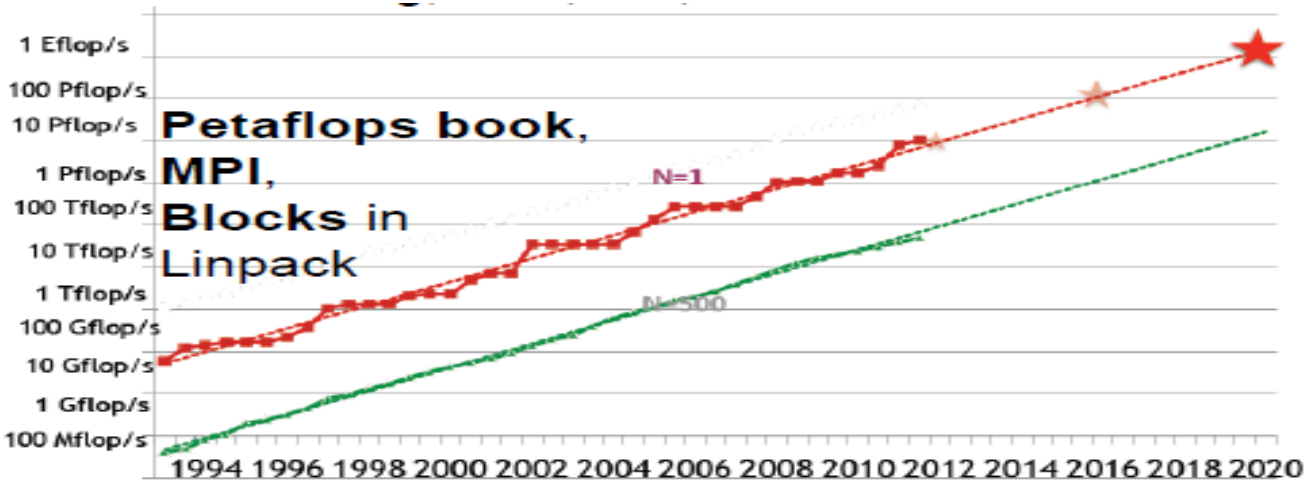
HPC only takes 20% market share



Big Data, AI, Internet Service…

What is the metric for emerging data center computing?

FLOPS defines HPC



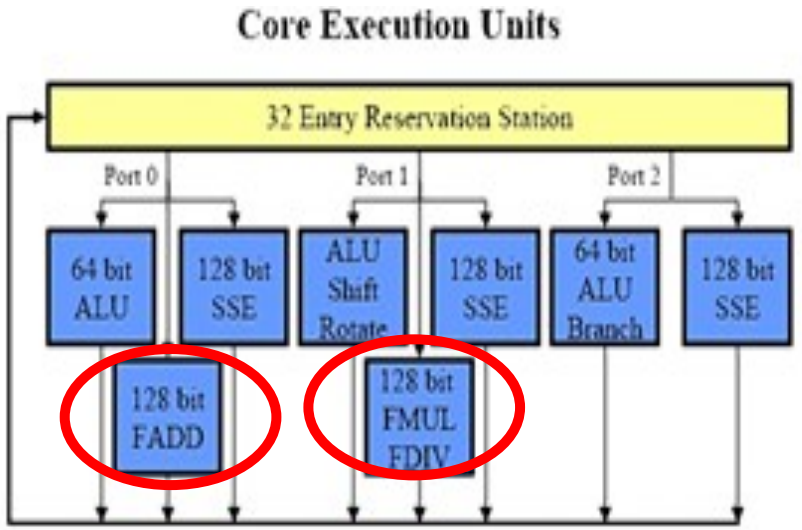
Jack Dongarra, On the Future of High Performance Computing: How to Think for Peta and Exascale Computing, SCI Institute, University of Utah, February 10, 2012,

1990: Gflops; 2000: Tflops; 2010: Pflops; 2020: Eflops

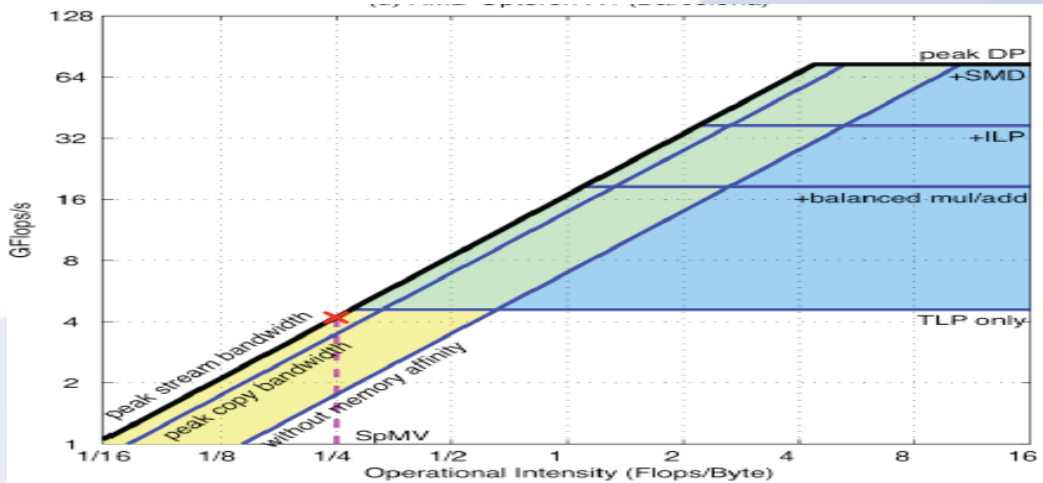
- **Independ with underlying implementation**
  - Fair evaluate and compare
- **Easily to be calculated at different level**
  - Facilitate co-design
- **Define the performance ceiling**
  - ***FLPOS = #(CPUs) ∗ #(Cores) ∗ Frequency ∗ #(FLOPs Per Cycle)***.
  - Help optimization

FLOPS: Measuring tools & Performance Model

- **Measuring tools -- HPL**
  - Fully utilize FPU
  - Obtain actual FLOPS
  - Efficiency
    - **100%*(*FLOPS*$_{Real}$ / *FLOPS*$_{Thoery}$)**



- **Roofline models**
  - *P = Min(PeakFLOPS,PeakMemoryBandwidthxOI)*



Is FLOPS still The metric in DC era?

Table 1: Experimental workloads

ID	Application	Software Stack	Input size
1	Sort	Hadoop,Spark,MPI	100GB data
2	Grep	Hadoop,Spark,MPI	100GB data
3	WordCount	Hadoop,Spark,MPI	100GB data
4	BFS	MPI	2E18 vertex
5	Read	Hbase	100 GB data
6	Write	Hbase	100GB data
7	Scan	Hbase	100 GB data
8	Select Query	Hive,Spark	100 GB data
9	Aggregate Query	Hive,Spark	100GB data
10	Join Query	Hive,Spark	100GB data
11	Nutch server	Hadoop	1000 req/s
12	PageRank	Hadoop,Spark,MPI	10E10 pages
13	Index	Hadoop	10E10 pages
14	Olio Server	Mysql	1000 req/s
15	Kmeans	Hadoop,Spark,MPI	100GB data
16	CC	Hadoop,Spark,MPI	2E18 vertex
17	Rubis Server	Mysql	1000 req/s
18	CF	Hadoop,Spark,MPI	2E18 vertex
19	Bayes	Hadoop,Spark,MPI	100 GB data

1) The average proportion of floating-point instructions **only takes 1%**
2) the average FLOPS efficiency is **only 0.1%**
3) The average CPU utilization is high as 63%.

FLOPS is inappropriate for evaluating DC computer systems.

Measurement: FLOPS+Linpack

	Xeon E5645	Xeon E5310	speedup
theoretical FLOPS	57.6GFLOPS	25.6GFLOPS	2.3
Real FLOPS	38.9 GFLOPS	19.1 GFLOPS	2.0
System Efficiency	68%	75%	-
Application View: DPS	110MB/S	52MB/S	2.1

	Xeon E5310	Atom D510	speedup
theoretical FLOPS	25.6GFLOPS	4.8GFLOPS	**5.3**
Real FLOPS	19.1 GFLOPS	3.2 GFLOPS	**6**
System Efficiency	75%	67%	**-**
Application View: DPS	40MB/S	11.5 MB/S	**3.5**

Cannot reflect the performance differences among different platform

Measurement: FLOPS+BigData Benchmark

	Xeon E5645	Xeon E5310	speedup
theoretical FLOPS	57.6 GFLOPS	25.6GFLOPS	2.3
Real FLOPS	0.02 GFLOPS	0.01GFLOPS	2.0
System Efficiency	**0.03%**	**0.04%**	-
DPS	110MB/S	52MB/S	2.1

	Xeon E5310	Atom D510	speedup
theoretical FLOPS	25.6GFLOPS	4.8GFLOPS	5.3
Real FLOPS	0.01GFLOPS	0.003 GFLOPS	**3**
System Efficiency	**0.04%**	**0.06%**	**-**
DPS	52MB/S	14.8 MB/S	**3.5**

Cannot reflect the system efficiency

Model: Roofline for DC

workloads	FLOPS	bandwidth	OS	Bottleneck
Sort	0.001	0.62	**0.002**	Memory Access
Grep	0.001	0.32	**0.003**	Memory Access
WordCount	0.001	0.46	**0.002**	Memory Access
Bayes	0.02	0.36	**0.05**	Memory Access
Kmeans	0.1	2	**0.05**	Memory Access

Operational Intensity (the total floating point operations divided by total memory access bytes) **is too small to reflect computing characteristic**

Requirements of the DC metric

- reflect the performance gaps between different systems
- support hardware and software co-design
- Easily measure the workload performance
- Decide the performance upper bound for a specific system

We present BOPS

- BOPS（Basic OPerations per Second）
  - BOPS is the average number of BOPs (Basic OPerations) completed each second.
- BOPs (Basic OPerations)
  - all of arithmetic, logical, comparing and array addressing operations for integer and floating point.

Basic operations	Arithmetic, bitwise operation, logical operations.
compare	>, <, >=, <=, =, !=
addressing	Array, pointer addressing

Normalization operations of BOPs

Operations	Normalized value
Add	1
Subtract	1
Multiply	1
Divide	1
Bitwise operation	1
Logic operation	1
Compare operation	1
One-dimensional array addressing	1
N-dimensional array addressing	N

operations not included

Category	Descriptions
Variable declaration	int i
Variable assignment	i=10
Type conversion	(int*) X
Branch call	goto, if else, Switch case
Loop call	for, while
Function call	Fun() call
Return	return command

BOPs Measurement: Source Code

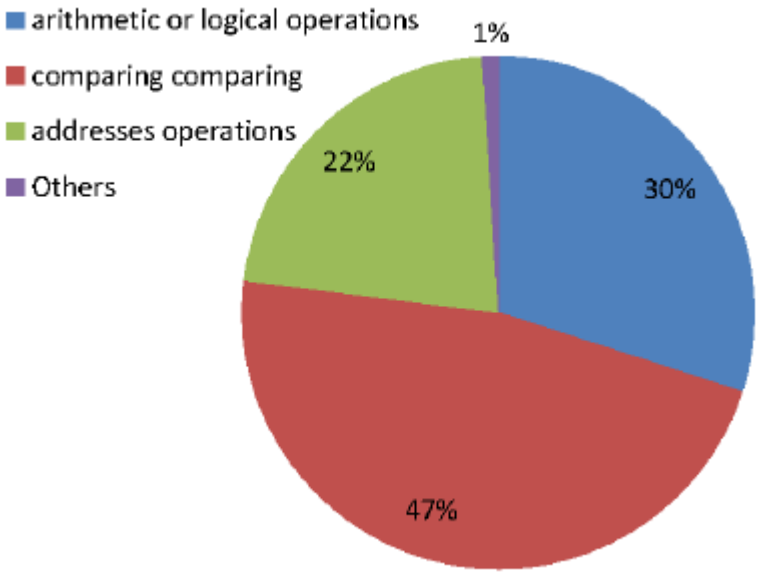
```
240 #ifndef __HAVE_ARCH_STRCMP
241 /**
242  * strcmp - Compare two strings
243  * @cs: One string
244  * @ct: Another string
245  */
246 #undef strcmp
247 int strcmp(const char *cs, const char *ct)
248 {
249         unsigned char c1, c2;
250
251         while (1) {
252                 c1 = *cs++;
253                 c2 = *ct++;
254                 if (c1 != c2)
255                         return c1 < c2 ? -1 : 1;
256                 if (!c1)
257                         break;
258         }
259         return 0;
260 }
261 EXPORT_SYMBOL(strcmp);
262 #endif
```

```
1    long    newClusterSize[100];
2    int     j;
3    for (j=0; j<100;j++)
4    {
5      newClusterSize[j]=0;
6    }
```

```
        movq    $0, ( %rax)
        addq    $8, %rax
        cmpq    %rdx, %rax
        jne     .L2
        movl    664(%rsp), %eax
        addq    $688, %rsp
        .cfi_def_cfa_offset 8
        ret
        .cfi_endproc
```

BOPs Measurement: Instruction level

- Approximate evaluation
  - **ins - branch_ins - load_ins - store_ins**
- Normally the deviation is usually < 0.08
  - |1−BOPSApproximateInstruction/BOPSSourceCode|



BOPS Benchmark

Workloads	Software Stacks	BOPs	Feature
Sort	C++ & MPI	529E9（10E8 entries）	IO-intensive（Integer）
Grep	C++& MPI	142E9（15GB）	Hybrid
WordCount	C++& MPI	179E9（15GB）	Hybrid
Bayes	C++& MPI	186E9(100MB)	CPU Intensive
Kmeans	C++& MPI	704E9（1GB）	CPU intensive（float-point）

Operation Type	Count
Inter	106E9
Float-point	0
compare	36E9
addressing	387E9
overall	529E9

Sort

Operation Type	Count
Int	34E9
Float point	0
compare	40E9
address	68E9
overall	142E9

Grep

Type	Count
Int	59E9
Float	0
Compare	20E9
Address	100E9
Overall	179E9

WordCount

Type	count
Inter	146E9
Float point	0.5E9
Compare	13.5E9
Address	26 E9
overall	186E9

Bayes

type	count
int	74E9
float	262E9
compare	81E9
address	287E9
overall	704E9

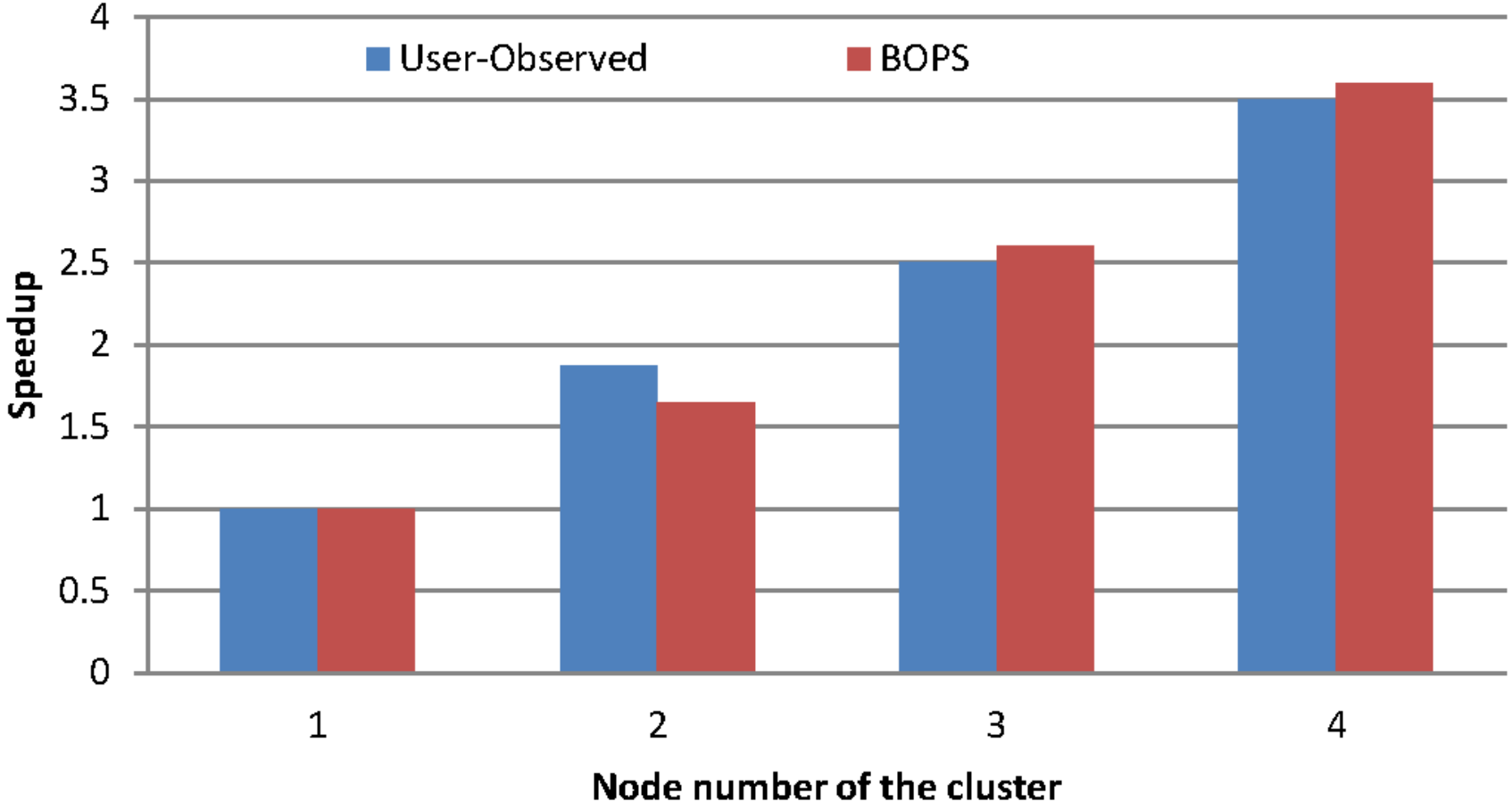
Kmeans

BOPS Vs. User-Perceivable metrics

	E5645	D510	Ratio
Peak GBOPS	86.4	12.8	6.7
Real average GBOPS	9.2	1.4	6.5
BOPS Efficiency	11%	10.9%	-
user-perceivable	110MB/S	14.8MB/s	7.4

BOPS can reflect the performance gaps between different platforms (6.5 v.s. 7.4)

Performance number report



both the BOPS and the user-perceivable metrics increase with a larger-scale cluster.

BOPS Vs FLOPS

	Xeon E5645	Xeon E5310	Speedup
theoretical FLOPS	57.6GFLOPS	25.6GFLOPS	2.3
Real FLOPS	38.9 GFLOPS	19.1 GFLOPS	2.0
Efficiency	68%	75%	-
Theory BOPS	86.4GBOPS	38.4GBOPS	2.3
Real BOPS	8.2 GBOPS	4.1 GBOPS	2.0
Efficiency	10%	11%	-
Application: DPS	110MB/S	52MB/S	2.1

	Xeon E5310	Atom D510	Speedup
theoretical FLOPS	25.6GFLOPS	4.8GFLOPS	5.3
Real FLOPS	19.1 GFLOPS	3.2 GFLOPS	6
Efficiency	75%	67%	-
Theory BOPS	38.4GBOPS	12.8GBOPS	3
Real BOPS	4.1 GBOPS	1.3GBOPS	3.2
Efficiency	11%	10%	-
Application: DPS	52MB/S	14.8 MB/S	3.5

BOPS for traditional workloads

	HPL	Graph500	Stream
GFLOPS	38.9	0.05	0.8
FLOPS efficiency	68%	0.04%	0.7%
GBOPS	41	12	13
BOPS efficiency	47%	18%	20%

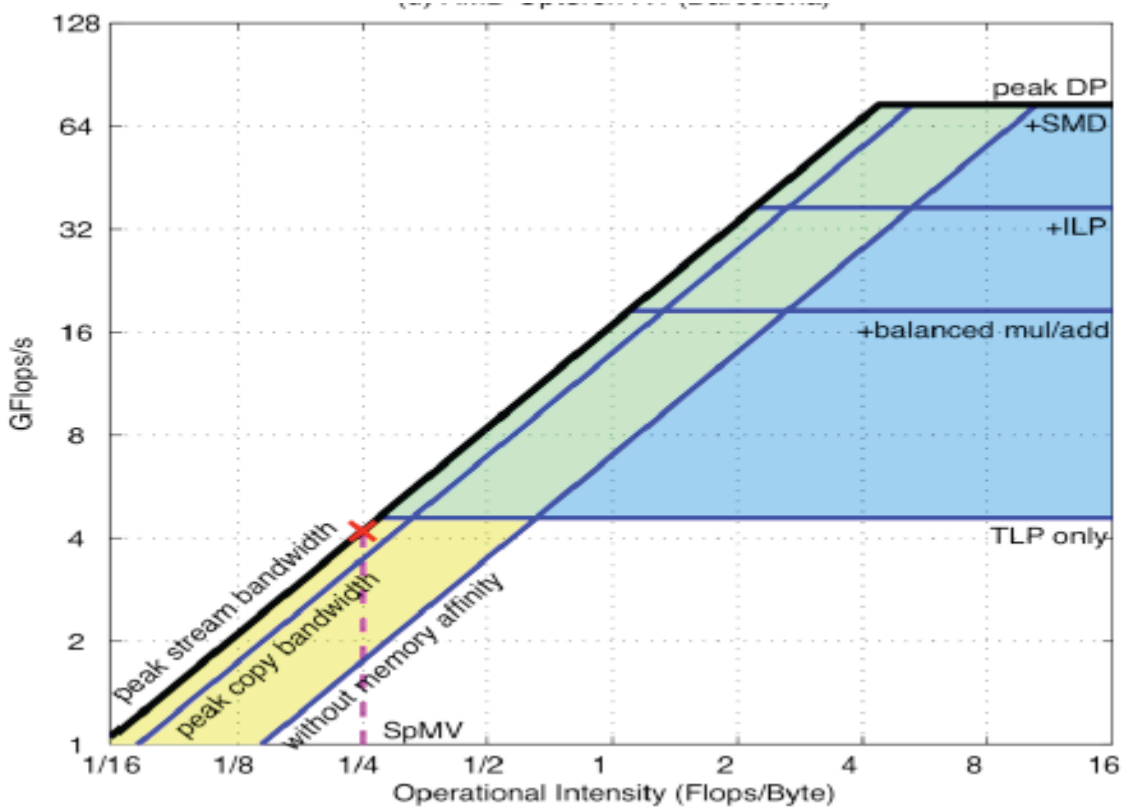
Also suited for traditional workloads

Discussion

- **Normalization**
  - Varied delays of different operations are not considered in the normalized calculations of basic operations, because delays can be extremely different in diverse micro-architecture platforms.

- The Efficiency of BOPS Measurement Tools
  - calculate the theoretical BOPS considering SIMD characteristics
    - none SIMD programming
  - Imbalance of operation instructions.
    - Executing port is not full used

Roofline Model



Operational Intensity: total number of floating point instructions divided by the total number of bytes of memory access

Samuel Williams, Andrew Waterman, and David Patterson, "Roofline: an insightful visual performance model for multicore architectures," Communications of the ACM, vol. 52, no. 4, pp. 65–76, 2009.

Model: Roofline

workloads	FLOPS	bandwidth	OI	Bottleneck
Sort	0.001	0.62	**0.002**	Memory Access
Grep	0.001	0.32	**0.003**	Memory Access
WordCount	0.001	0.46	**0.002**	Memory Access
Bayes	0.02	0.36	**0.05**	Memory Access
Kmeans	0.1	2	**0.05**	Memory Access

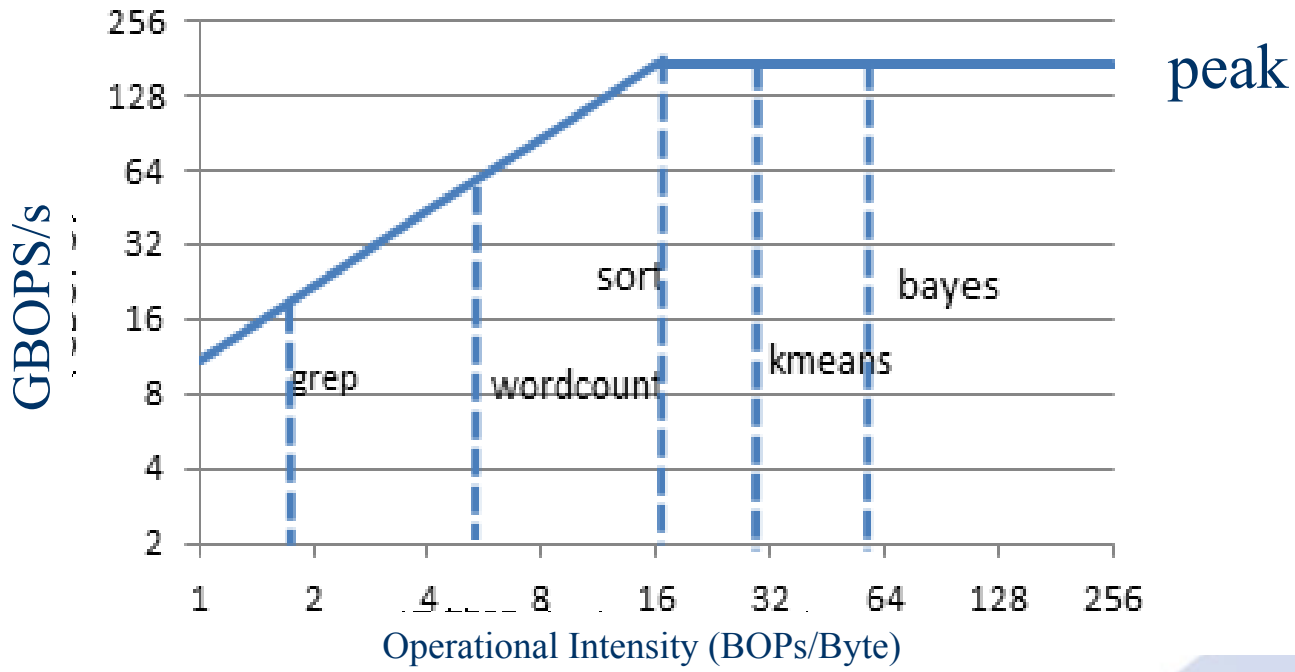
Operational Intensity (the total floating point operations divided by total memory access bytes) **is too small to reflect computing characteristic**

屋顶模型在大数据计算中的不足

负载	FLOPS	平均带宽	计算强度	瓶颈分析
Sort	0.001	1.7	**0.002**	访存
Grep	0.001	4.8	**0.003**	访存
WordCount	0.001	1.2	**0.002**	访存
Bayes	0.002	0.4	**0.005**	访存
Kmeans	0.1	0.5	**0.2**	访存

计算强度太小！

Roofline for DC computing



BOPs Operational Intensity: total number of floating point and integer instructions divided by the total number of bytes of memory access

BOP： integer + float point

BOPS Peak: 172.8GBOPS; Memory Bandwidth Peak: 15GB/S

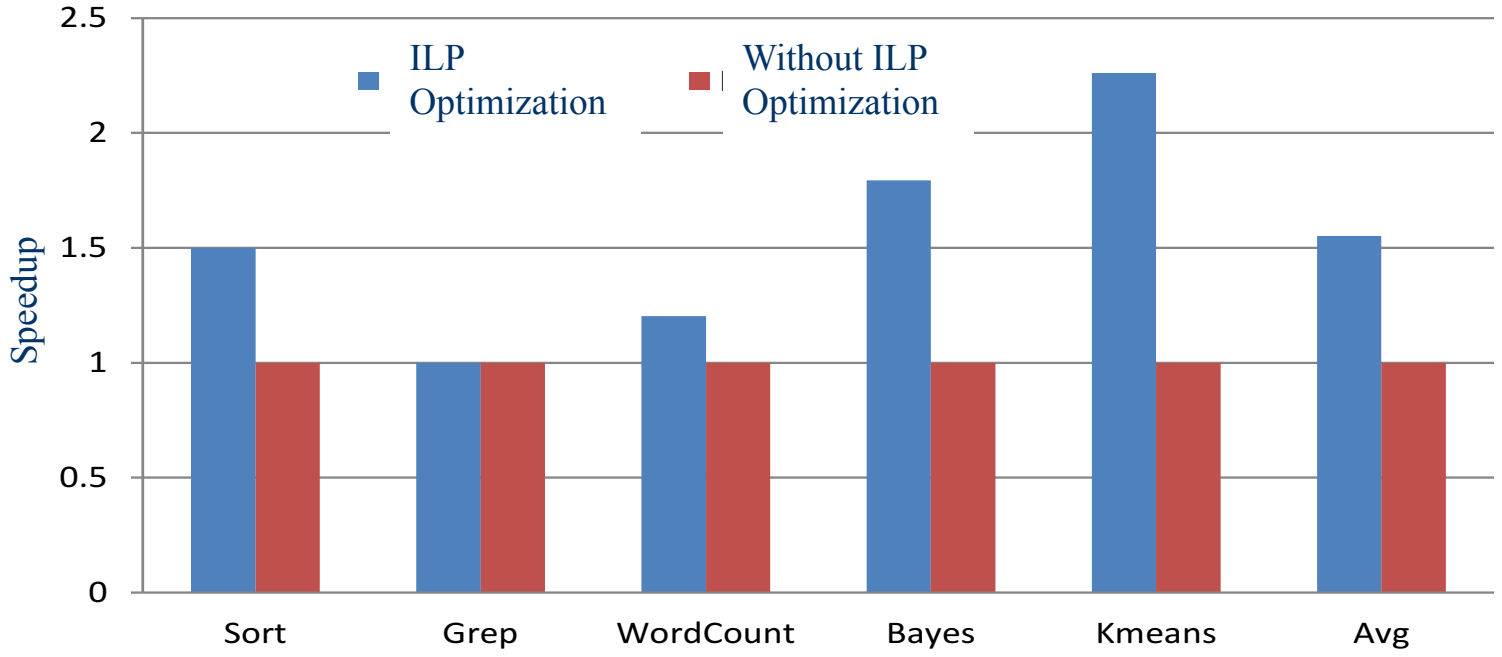
Dual Intel Xeon E5645 Platform

FLOPS roofline V.S. BOPS roofline

workloads	Real FLOPS	bandwidth	OI	Bottleneck
Sort	0.001	1.7	0.002	Memory Access
Grep	0.001	4.8	0.003	Memory Access
WordCount	0.001	1.2	0.002	Memory Access
Bayes	0.002	0.4	0.005	Memory Access
Kmeans	0.1	0.5	0.2	Memory Access

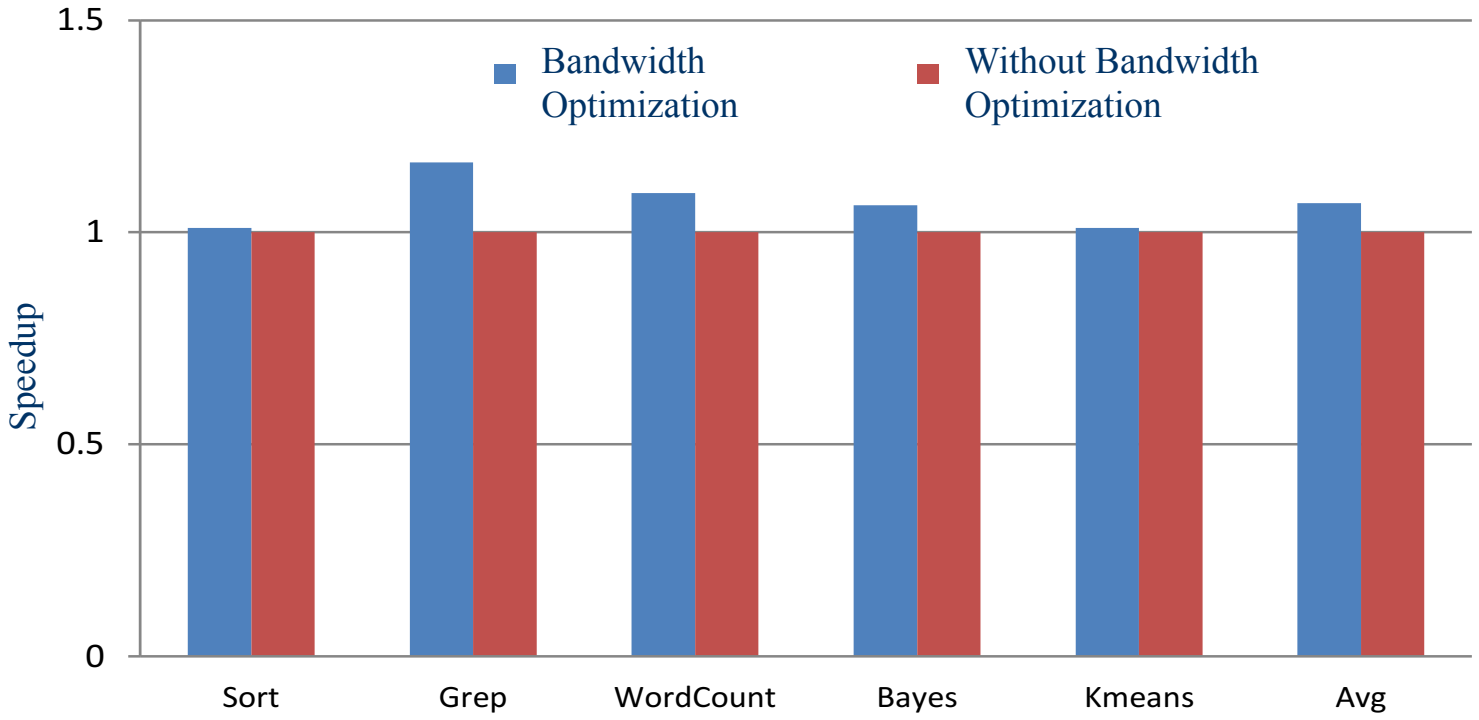
workloads	Real BOPS	Bandwidth	OI	Bottleneck
Sort	14.2	1	14	Calculation
Grep	9.2	4.8	1.9	Memory Access
WordCount	6.9	1.1	6.3	Memory Access
Bayes	10.2	0.1	102	Calculation
Kmeans	5.0	0.2	25.1	Calculation

Compiler optimization



BOPs of Sort, Bayes, and Kmeans has significant improve as 50%、78%, and 120%

Bandwidth Optimization



BOPS of Grep and WordCount has improved by 16% and 10%, respectively

Different Software Stacks

MPI	Real BOPS	Bandwidth	OI	Bottleneck
Sort	21.3	1.7	12.5	**Calculation**
Grep	9.2	5.6	1.6	**Memory Access**
WordCount	8.3	1.2	6.9	**Memory Access**
Bayes	18.3	0.4	45.7	**Calculation**
Kmeans	11.3	0.5	22.6	**Calculation**

Hadoop	Real BOPS	Bandwidth	OI	Bottleneck
Sort	6.3	1	1.3	**Memory Access**
Grep	7.9	1.6	4.6	**Memory Access**
WordCount	4.5	2.4	1.8	**Memory Access**
Bayes	8.7	2.2	3.9	**Memory Access**
Kmeans	7.7	1.8	4.2	**Memory Access**

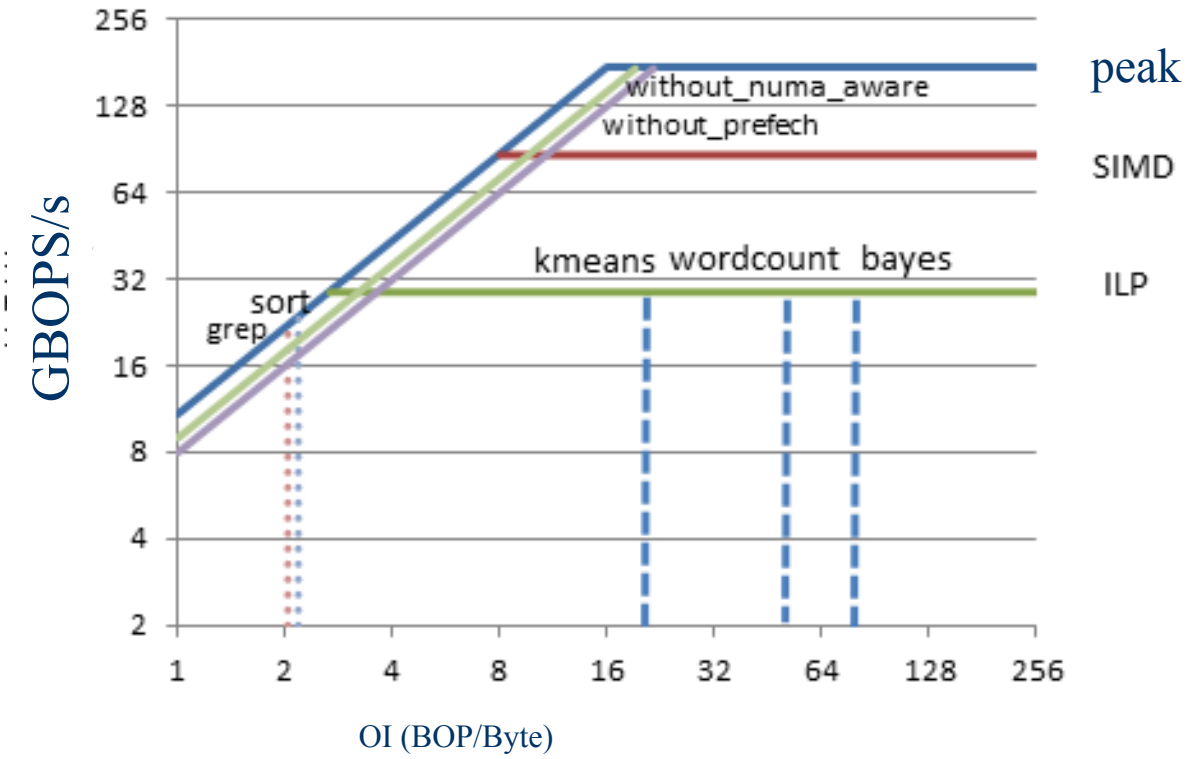
Different Platform

E5645	Real BOPS	Bandwidth	OI	Bottleneck
Sort	21.3	1.7	12.5	Calculation
Grep	9.2	5.6	1.6	Memory Access
WordCount	8.3	1.2	6.9	Memory Access
Bayes	18.3	0.4	45.7	Calculation
Kmeans	11.3	0.5	22.6	Calculation

E5310	Real BOPS	Bandwidth	OI	Bottleneck
Sort	8.7	0.8	10.8	Calculation
Grep	1.4	2	0.7	Memory Access
WordCount	1.4	0.4	3.5	Memory Access
Bayes	4.6	0.1	46	Calculation
Kmeans	4.7	0.2	23.2	Calculation

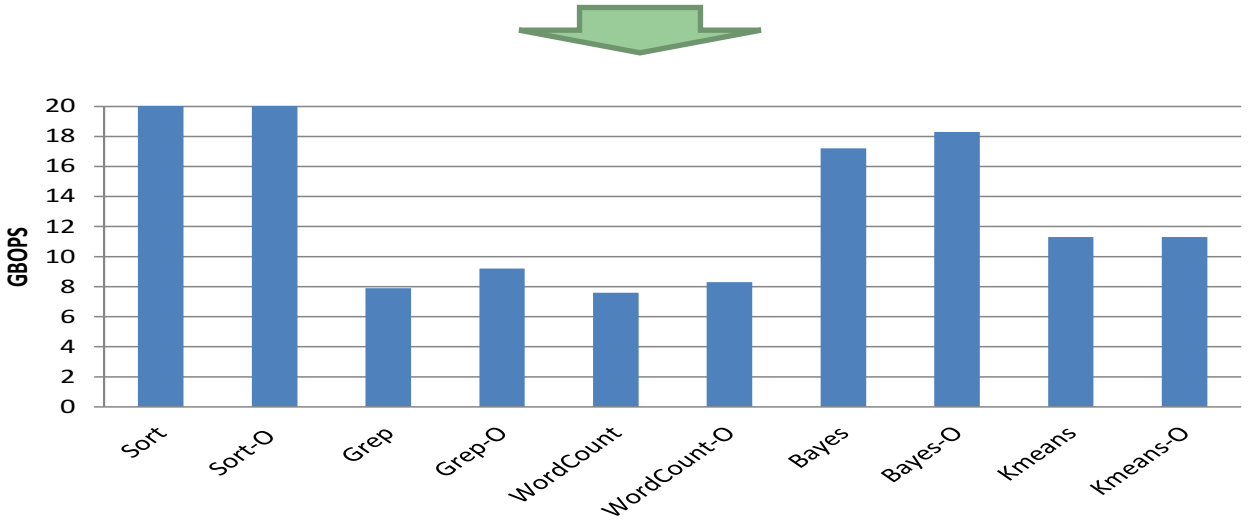
D510	Real BOPS	Bandwidth	OI	Bottleneck
Sort	2.6	0.4	6.5	Memory Access
Grep	0.6	0.2	3	Memory Access
WordCount	0.5	0.1	5	Memory Access
Bayes	1.1	0.1	11	Memory Access
Kmeans	1.5	0.2	7.5	Memory Access

System Optimization



Memory Bandwidth Optimization

Method	Bandwidth
none	13.2GB/s
NUMA Support	14.2GB/s
Hardware Pre-fetch	14.8GB/s
adjacent cache line prefetch	14.8GB/s
dcu Streamer prefetch	14.8GB/s
dcu Streamer prefetch	14.8GB/s



In Summary

- FLOPs and corresponding roofline model is not appropriate for new DC computing
  - Very little floating point instruction
  - FLOPS efficiency is too low for DC workloads
  - Deviation from the user-perceivable metrics

- BOPs and Its Roofline model, measuring Tools

Find the full Technical Report below:

http://prof.ict.ac.cn/BigDataBench/wp-content/uploads/2018/BOPS-2018.pdf

Thank you!