\Orchestrating a brighter world NEC

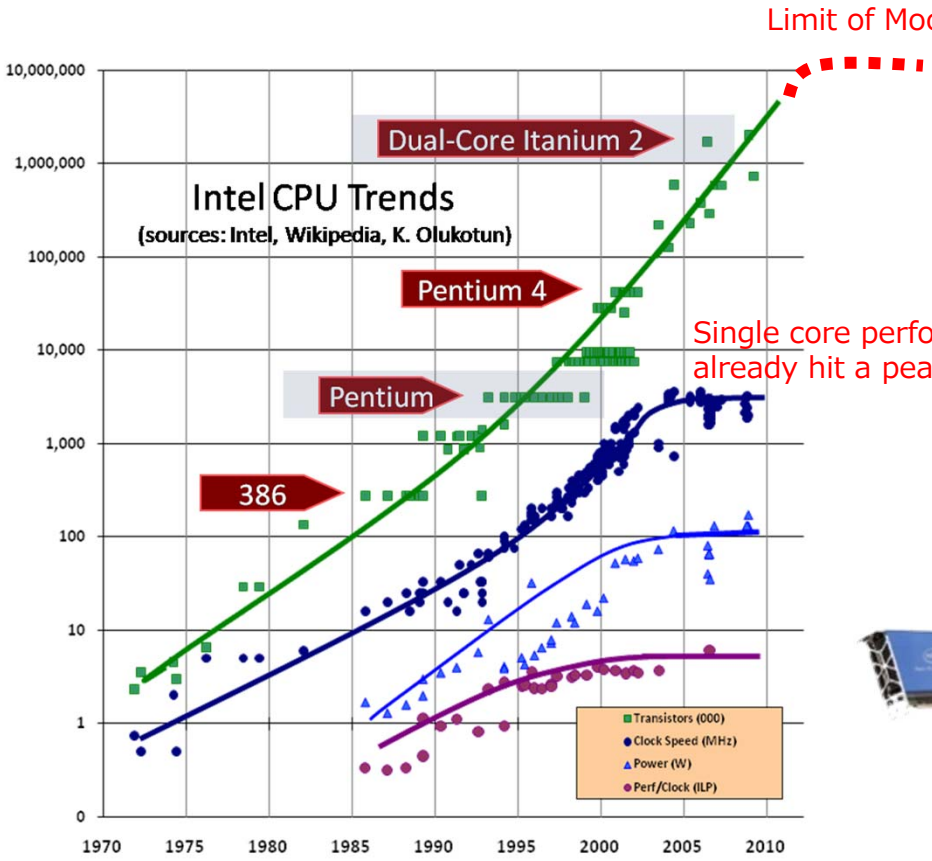
You Could Speed up Your Program by Using a Vector Processor.

Yuichi Nakamura
General Manager of System Platform Research Laboratories
NEC Corp.

Computing Trend

Limit of Moor's law:

more transistors != higher performance



http://www.gotw.ca/publications/concurrency-ddj.htm

Single-core(increasing Hz)

Power wall

Multicore/Manycore
(increasing # of cores)

Dark Silicon

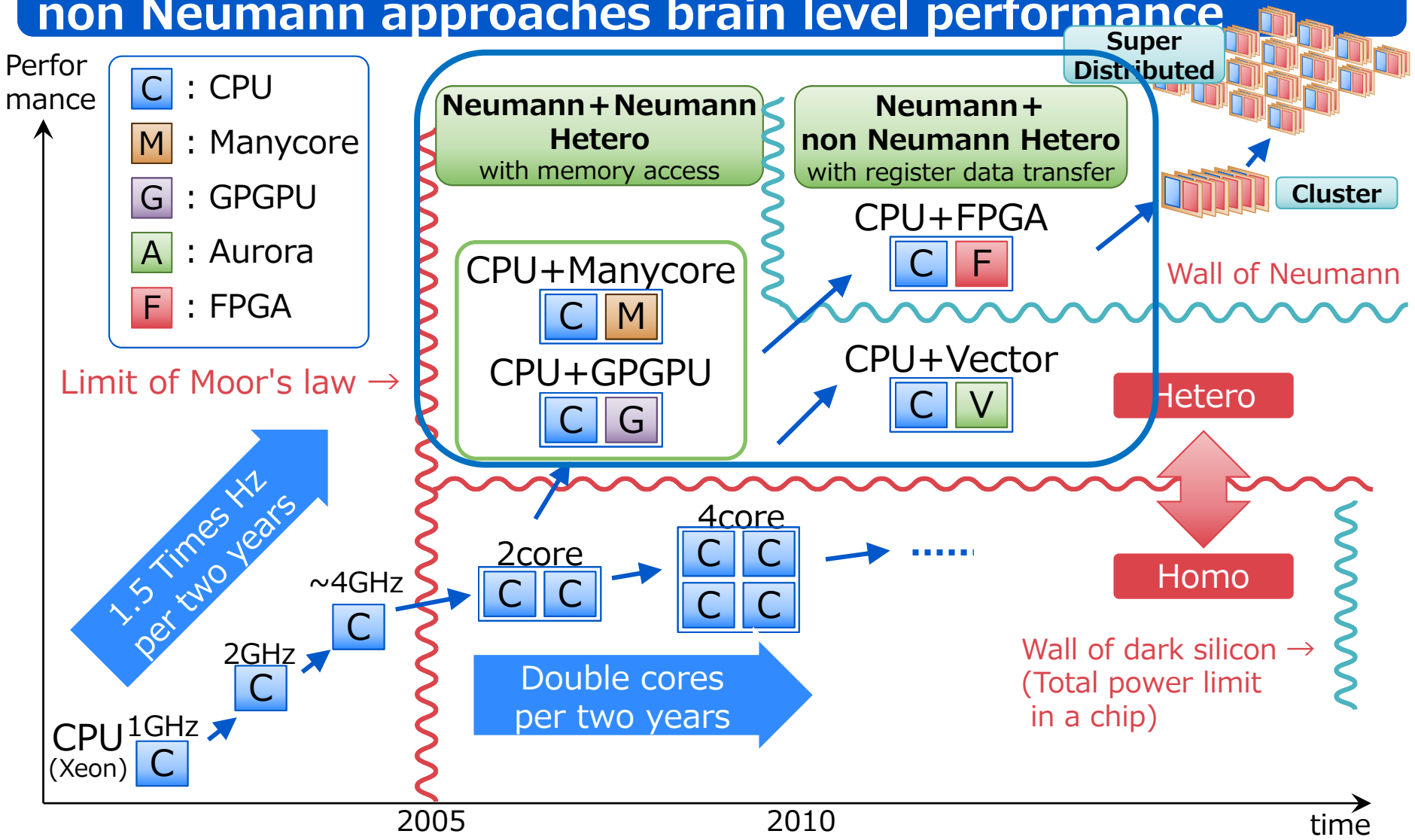
Hetero Computing
(use of accelerators)

http://newsroom.intel.com/docs/DOC-3126

http://www.nvidia.com/object/io_1238654717841.html

Hetero Computing

Hetero of general-purpose Neumann and high efficient non Neumann approaches brain level performance



How to Design Hetero Computing

Select the best processing engine (accelerator) for the target application.

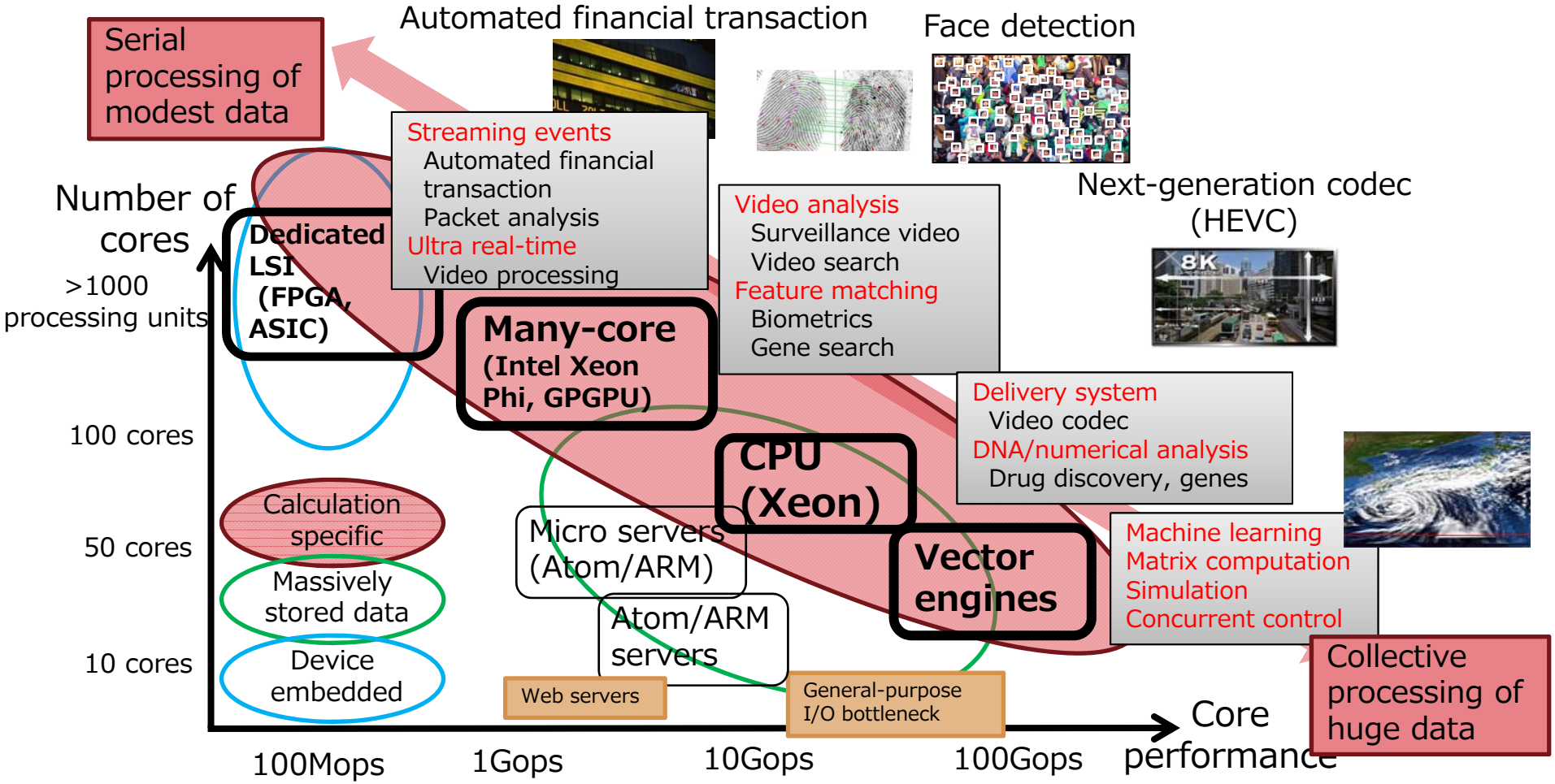
- The most important thing is to know the advantages and the disadvantages of the processing engines.
- The length of thread, the length of data size, the kind of processing, and etc.
- How to implement the application to the processing engine

Combine the selected processing engines (not presented today)

- Communication architecture
- Scheduling

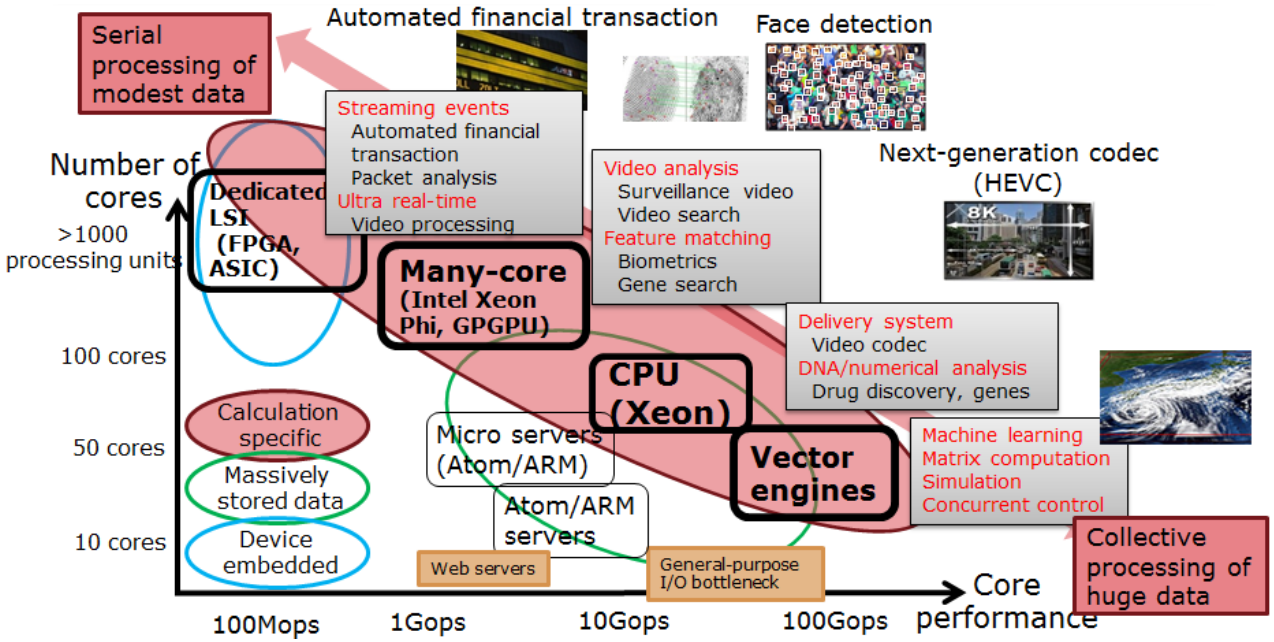
Selection of optimal processing engines according to target applications

The selection of processing engines according to the required processing characteristics shall be optimized to further enhance the computing performance



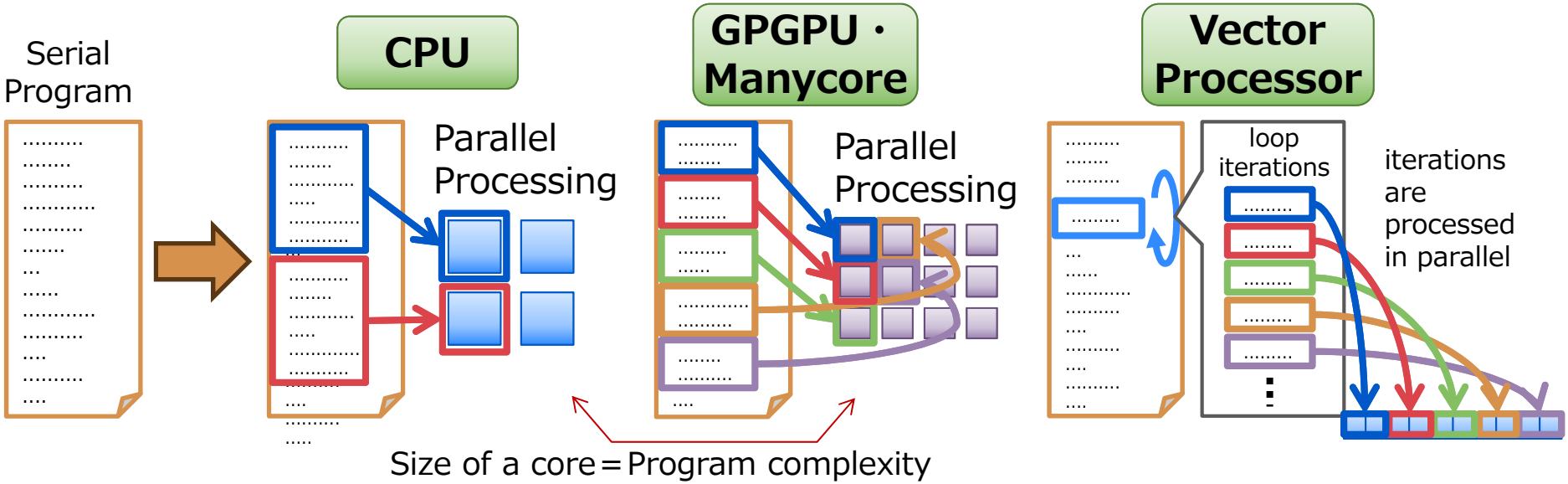
Hetero Computing

- Combined use of multiple types of engines for higher performance beyond limitation (performance and power consumption) of homogeneous computing
- Engines for hetero computing
  - von Neumann Architecture
    - CPU
    - GPU
    - Manycore
    - Vector Processor
  - non von Neumann Architecture
    - FPGA



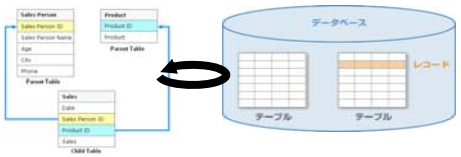
Software View of von Neumann Architecture

CPU/GPGPU/Manycore/Vector : Serial program is divided to fit the processing units



Some Complex ⟷ Selection of the best engine for higher application performance ⟷ Many Simple

Transaction Processing



Biometrics

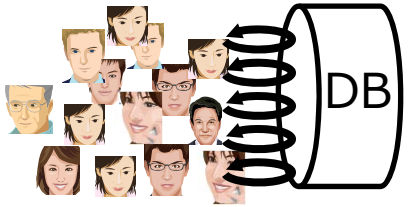
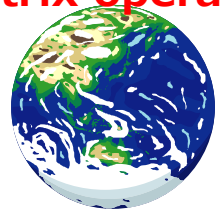


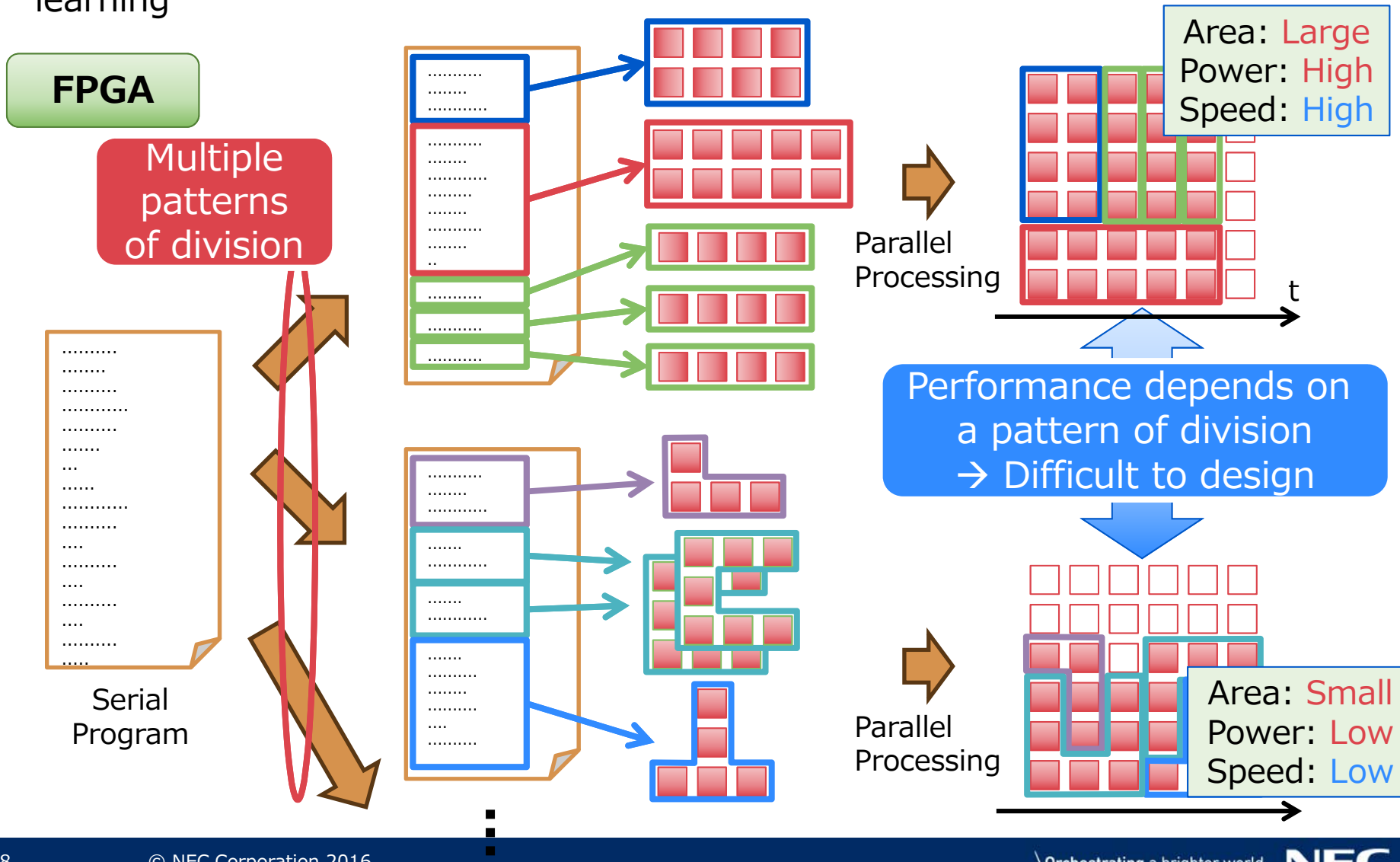
Image Processing

Physics Simulation (Matrix operation)



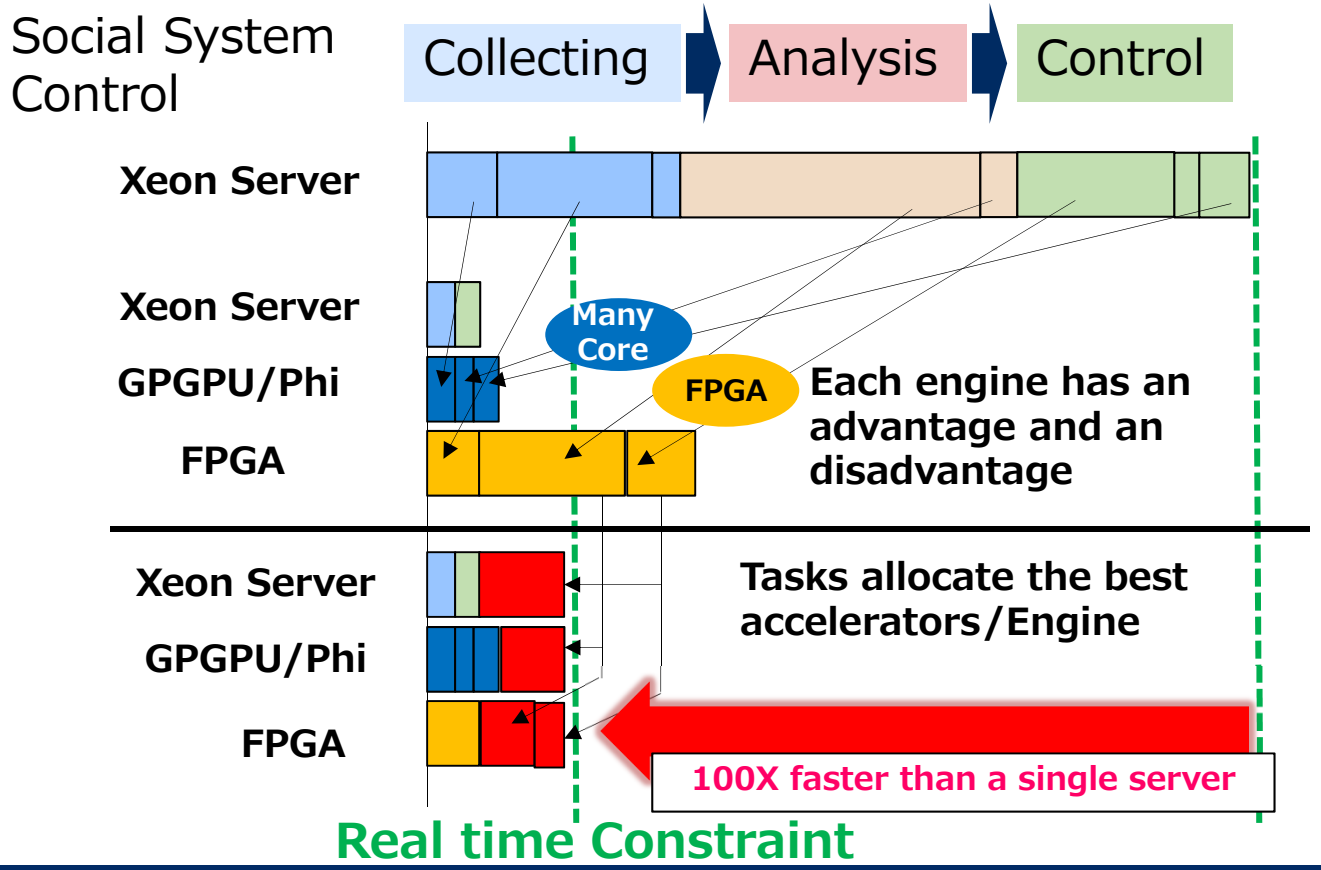
SW view of non von Neumann architecture

- FPGA : flexible processing unit → difficult to use
  - MS, JP-Morgan, and Baidu use FPGA for bigdata processing and machine learning



Combination of Processing Engines

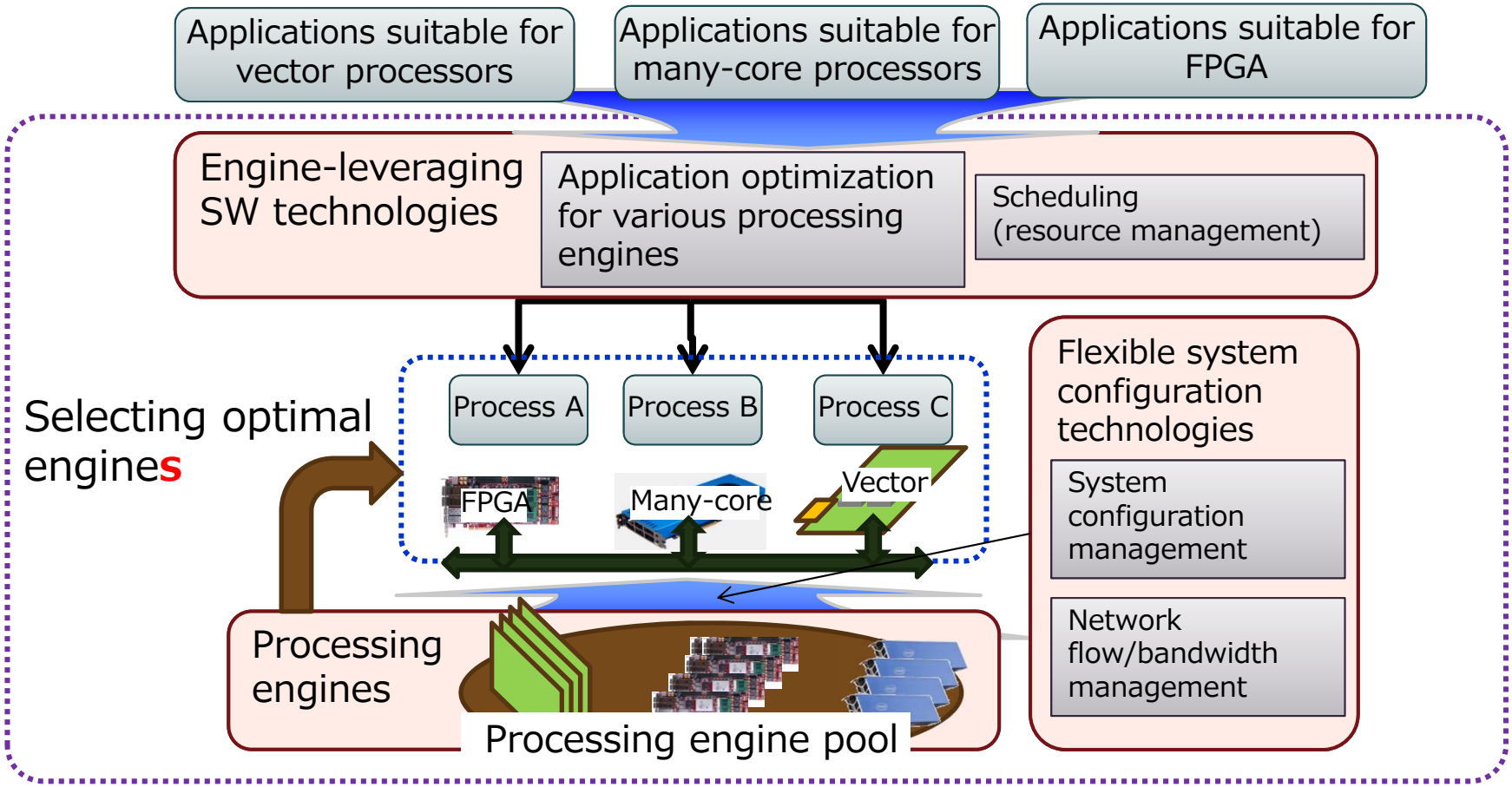
- Social system is consisted of various applications.
- Each processing engine has an advantage and disadvantage according to application/task/process/function.
- One engine cannot solve the problem.
- Use the best processing engine and combine them.



Example:
Video severance

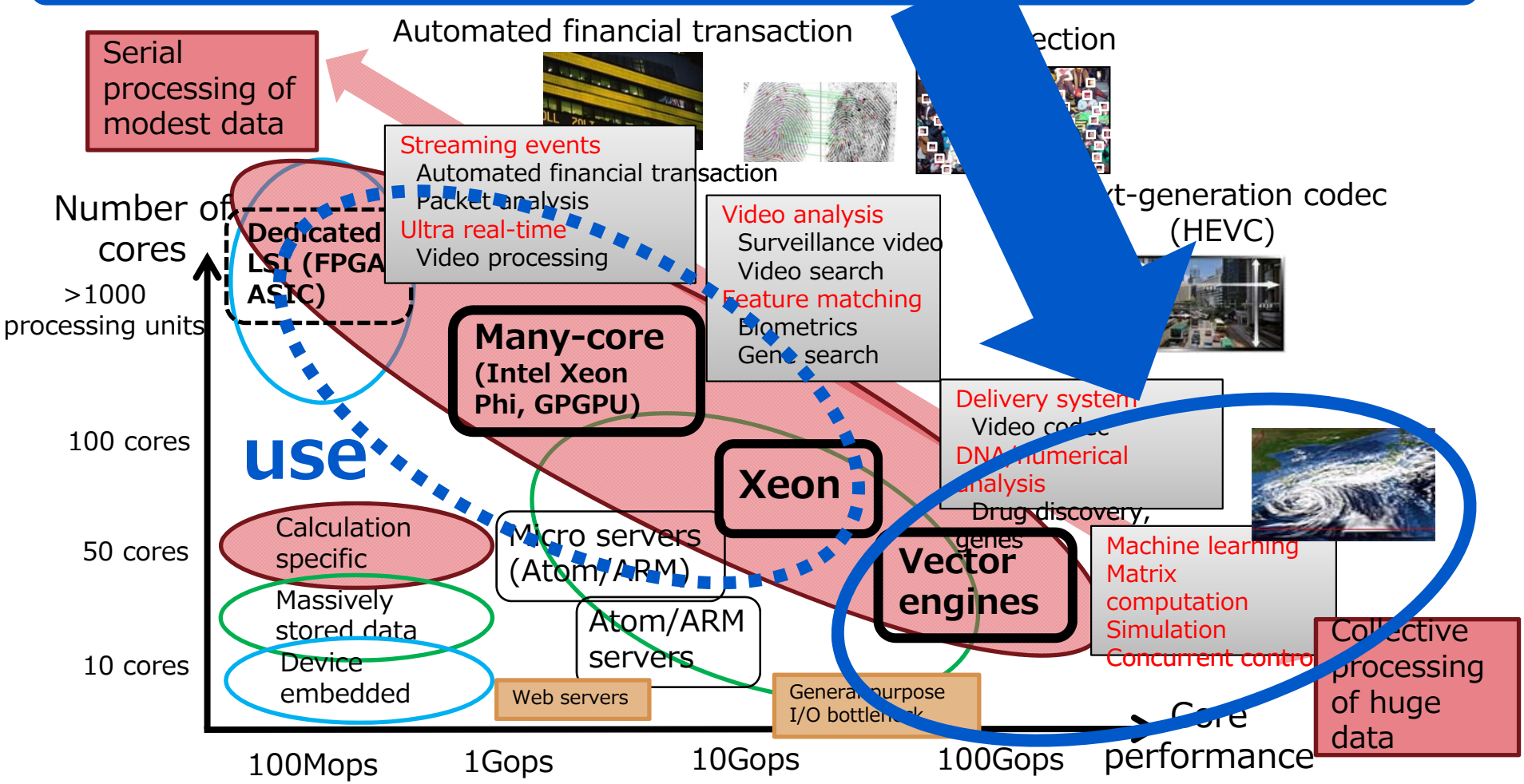
Processing Engine Pool for Hetero Combination

The performance is maximized by optimizing the combination of heterogeneous processing engines with software technologies to achieve the required processing capability at low cost.



Vector Processor

NEC is developing vector processor for super computer over 30 years for continuous needs of HPC from industries



History and Technical Evolutions of Super Computer

NEC has always provided the high sustained performance by Vector Super-Computer SX series.

Support >1000 nodes
ECO

upport
itilane IXS

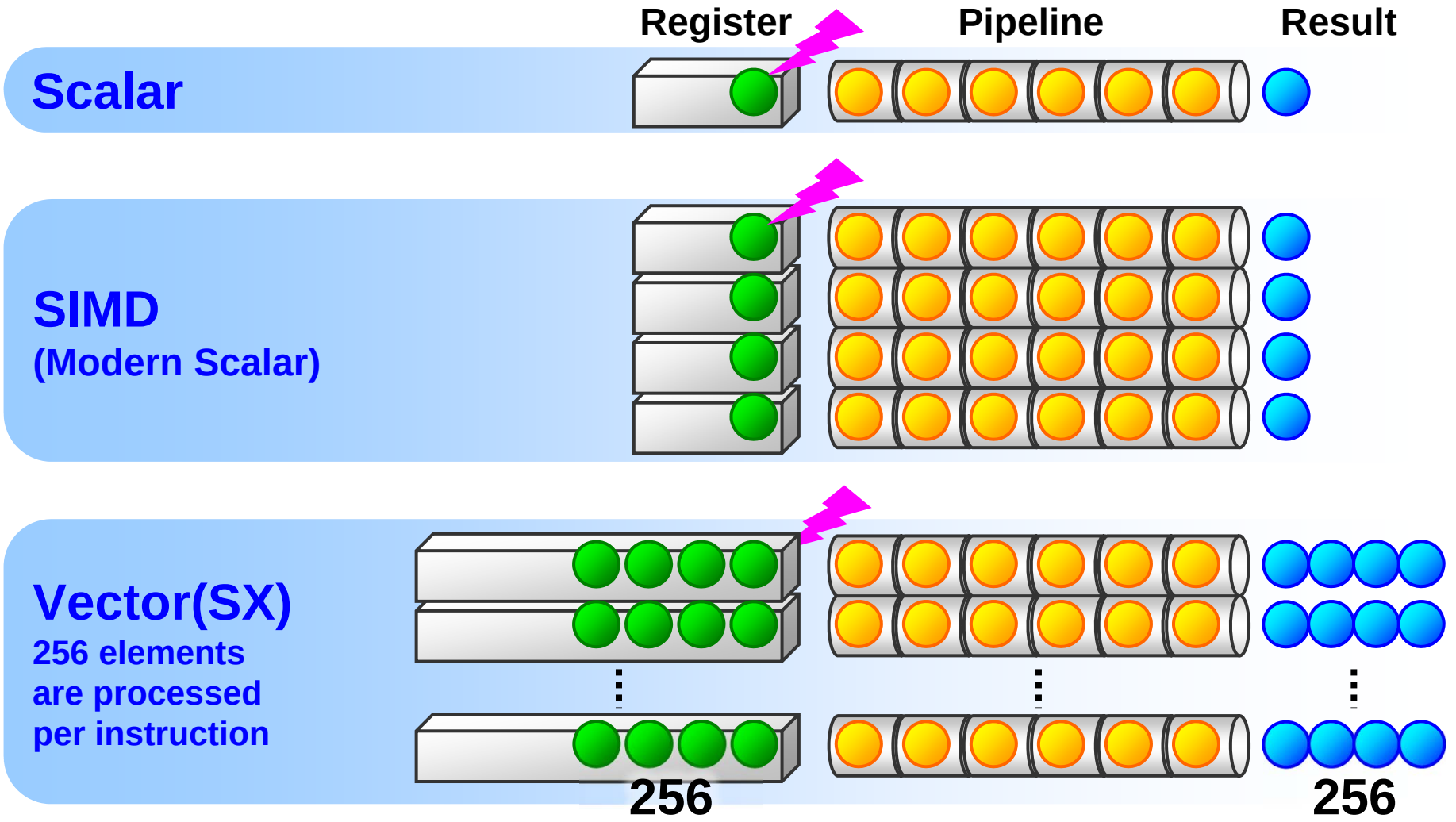
World No.1 Super Computer (Top500 2002~2004)

Gordon Bell Award (2002)

SX-2 Water Cooling

1990 2000 2010

Vector Processor



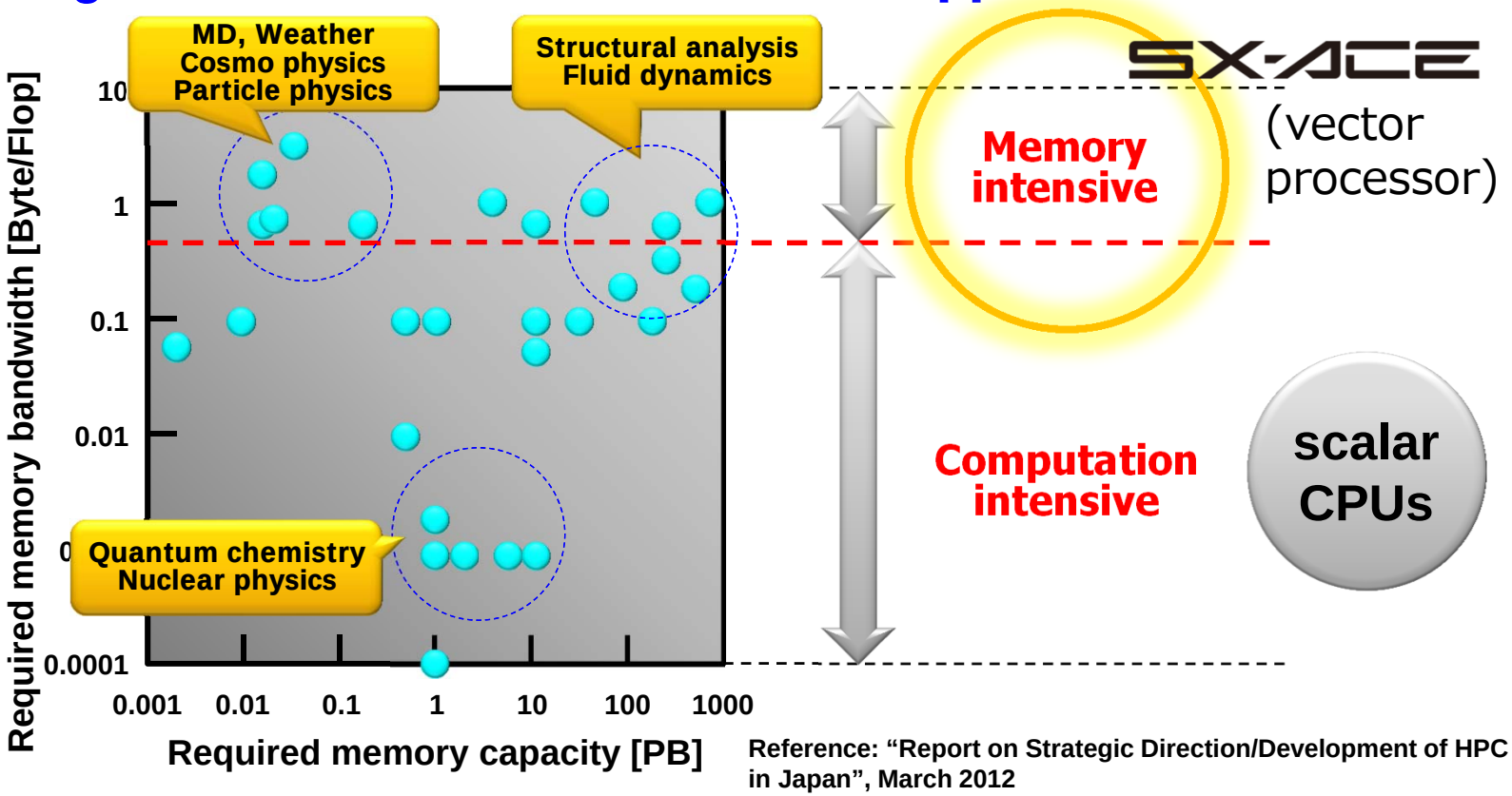
Vector is efficient to fill pipelines and to hide latencies

Required Byte/Flop in Real Applications

According to Japanese Government (MEXT) working group report for a wide variety of strategic segment applications, diverse characteristics are observed.

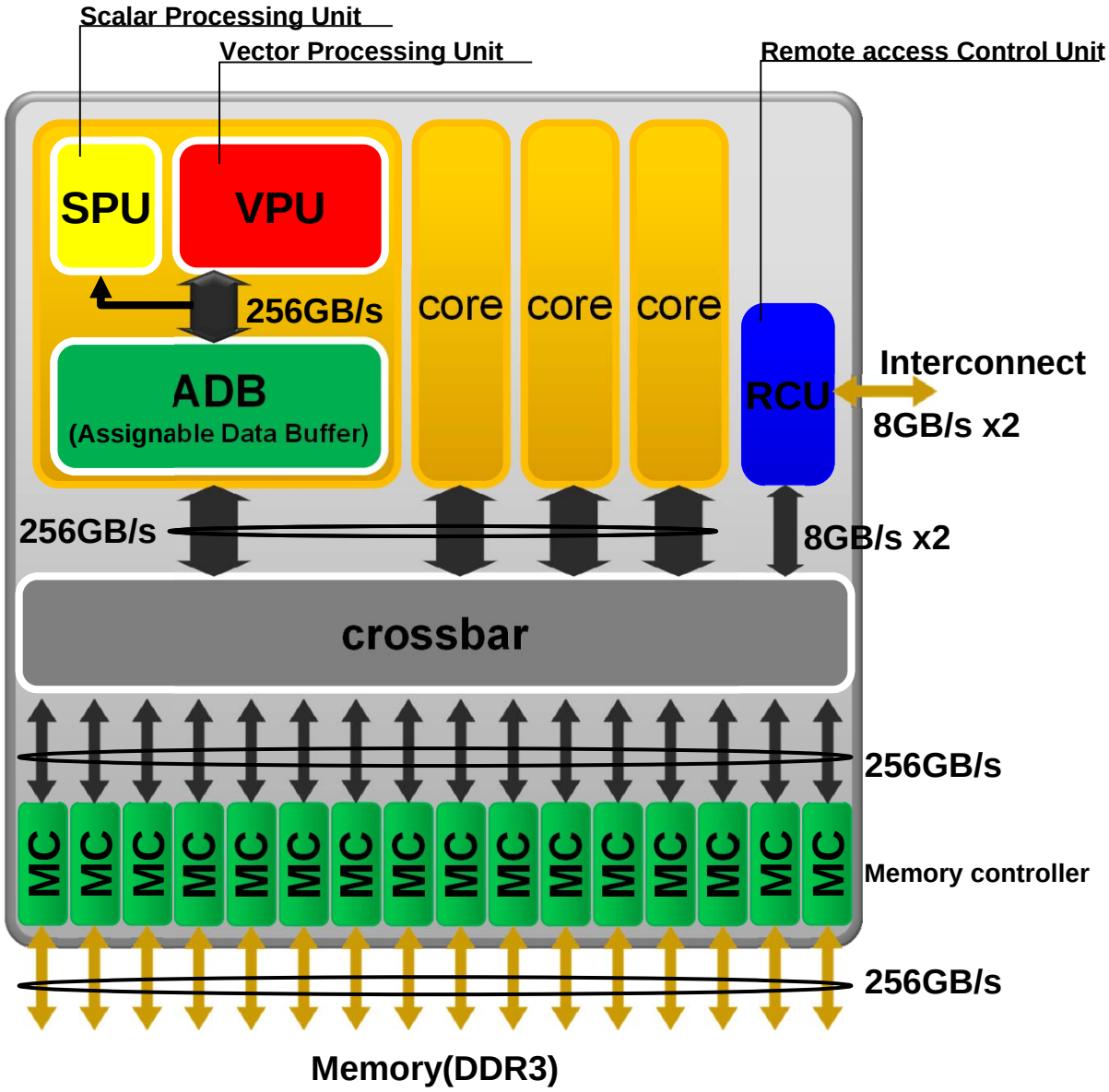
MEXT: Ministry of Education, Culture, Sports, Science & Technology

B/F requirement from each application differs greatly.
Any single architecture cannot cover all application areas.



Reference: "Report on Strategic Direction/Development of HPC in Japan", March 2012

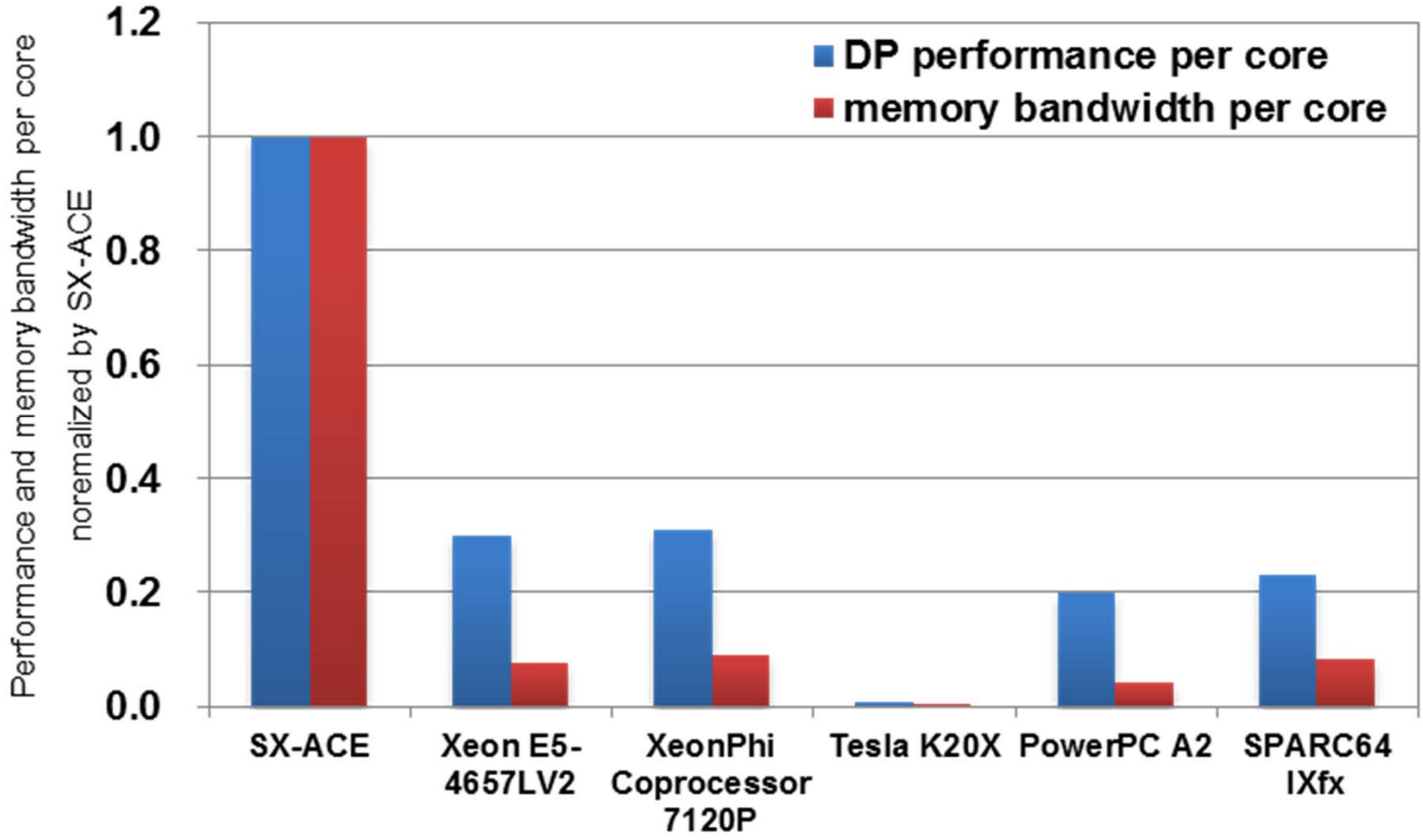
Processor Overview(SX-ACE)



CORE	
Architecture	Vector
Clock Frequency	1.0GHz
SPU decode rate	4 instructions
VPU Performance	64GFlops
ADB size	1MB
ADB bandwidth	256GB/s
Memory bandwidth	64GB/s~ 256GB/s
Core Byte/Flop	1.0 ~ 4.0
CPU	
Cores	4
Performance	256GFlops
Memory bandwidth	256GB/s
CPU Byte/Flop	1.0
Memory capacity	64GB

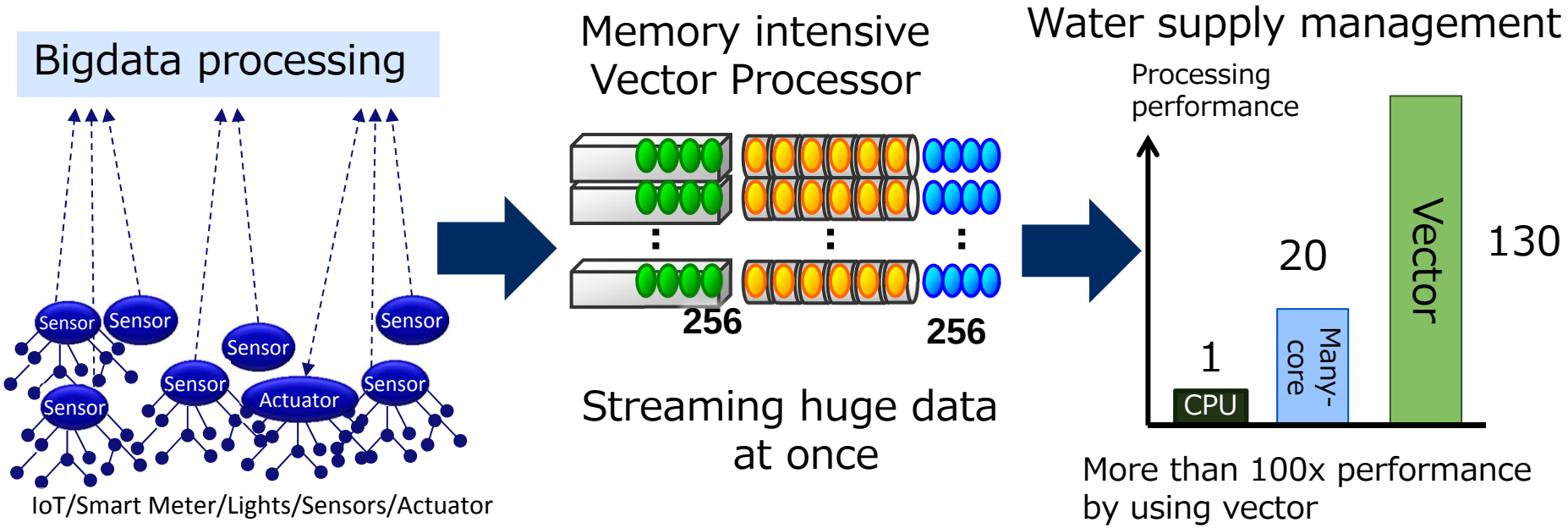
Single Core Comparison

The SX-ACE core can provide the world top-level performance and the largest memory bandwidth



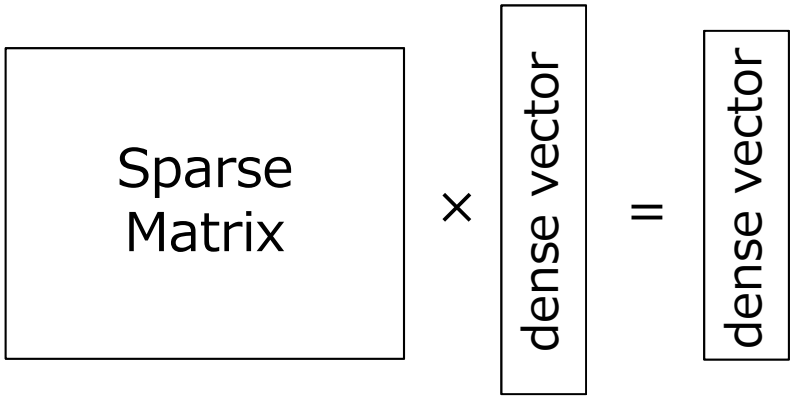
Bigdata Processing with Vector Processor

- Vector processor has a great advantage to process streaming huge data continuously.
- Bigdata which collects many terminals, consists of “streaming huge data”.
  - All data unit are applied the similar processing.
- We confirmed our vector processor can achieve significant speed up compared with CPUs or Many core processors, on several bigdata processing.



Vector Processing for Bigdata (1/2)

- Recently, machine learning (ML) is becoming important in Big Data analytics
- Most ML algorithms can be written as “matrix operation”, which can be executed efficiently with vector architecture
- Especially, large scale ML tends to use “sparse matrix”, which vector architecture is very good at
  - e.g. sparse matrix vector multiplication (SpMV)



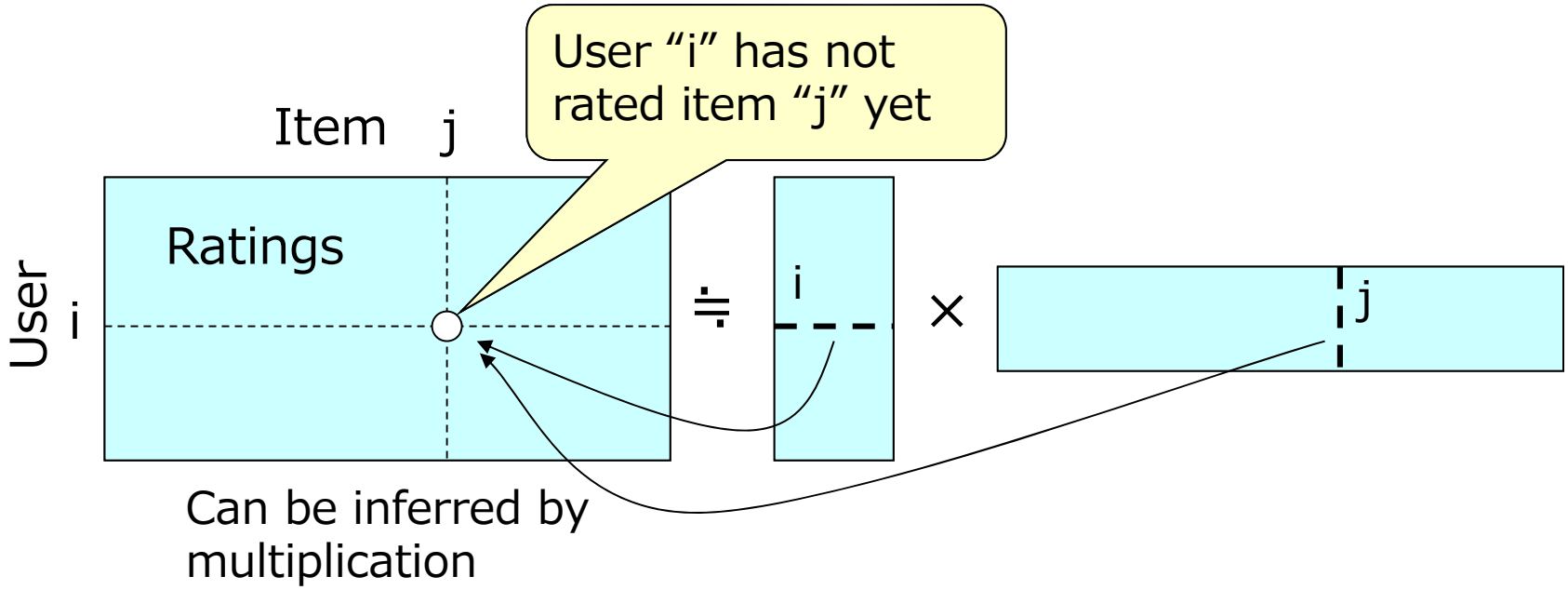
Vector Processing for Bigdata (2/2)

- On the other hand, users in Big Data field are not accustomed to low layer programming like MPI
  - Use middleware like “Spark”
  - Spark is written in Scala that runs on Java Virtual Machine; it is difficult to execute it on vector architecture efficiently

- We prototyped a middleware that runs on “Vector Processor#
  - Provides interface like Spark
- We did preliminary evaluation using an ML algorithm
  - Showed its effectiveness by the evaluation on “Vector Processor”

Example of ML Algorithm (1/3): Recommender System

- Create a matrix: rating (e.g. purchase) of item j by user i
  - Sparse matrix
- Estimate evaluation value that is not evaluated yet; recommend item of high evaluation value



Example of ML (2/3): Prediction of CTR (click through rate)

Predict if advertisement will be clicked or not by history data

- Use the history as training data
- Algorithm: logistic regression, support vector machine, etc.

Training data

- e.g. URL, referrer URL, banner size, category, user info, etc.
- Convert them into matrix for the input of ML algorithm

Example of conversion: one-hot encoding

- In the case of URL, each URL are assigned to different dimension
- Sparse matrix

yahoo	google	Bing	Baidu	hatena	…
1					
			1		
		1			

Example of ML (3/3): Document Analysis

- Typically document-term matrix is created
  - column: word, row: document, value: number of occurrence
  - Sparse matrix
- Various operations on this matrix:
  - Clustering: gather similar document (e.g. K-means)
  - Topic analysis: analyze “topic” of this document (e.g. LSA, LDA)
    - LSA (Latent Semantic Analysis) can be implemented by singular value decomposition (SVD) of the matrix

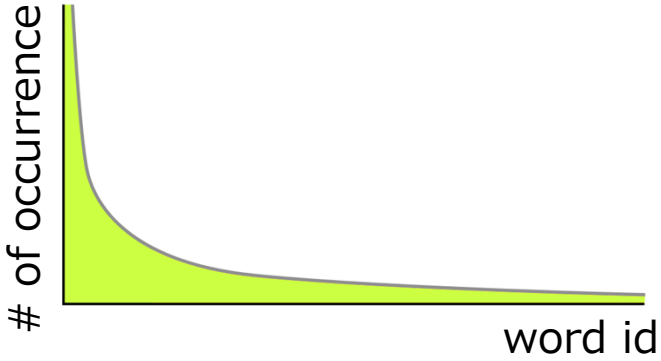
	this	is	a	pen	I	am	boy
this is a pen	1	1	1	1			
I am a boy			1		1	1	1
…							

Optimization of SpMV for ML (1/5)

- As stated before, large scale ML tends to use sparse matrix
  - SpMV often becomes the kernel operation of sparse matrix

- Here, sparse matrix of ML typically follows “power law”
  - e.g. document-term matrix
    - column: word, row: document, value: number of occurrence
    - some words (e.g. “a”) appears much more frequently than others

	this	is	a	pen	I	am	student
this is a pen	1	1	1	1			
I am a student			1		1	1	1
	…						

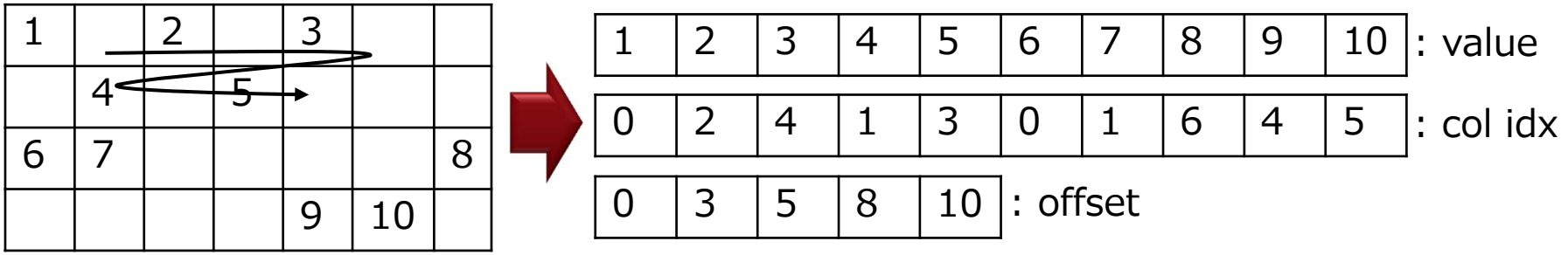


Need for optimized SpMV for such matrix

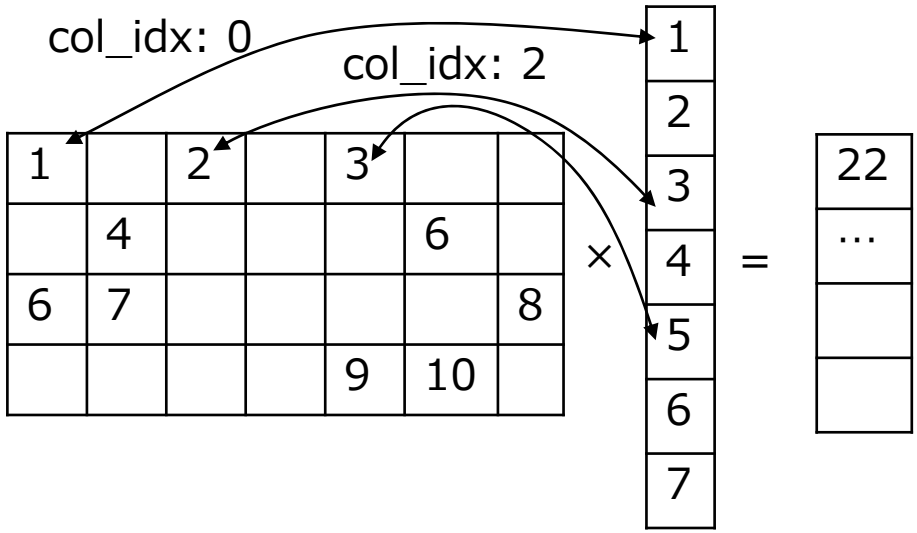
Optimization of SpMV for ML (2/5)

Existing matrix format (1): Compressed Row Storage (CRS)

- store the non-zero elements contiguously in row major order
- “column index” and “offset” of each row is also stored



- SpMV can be implemented using dot-product

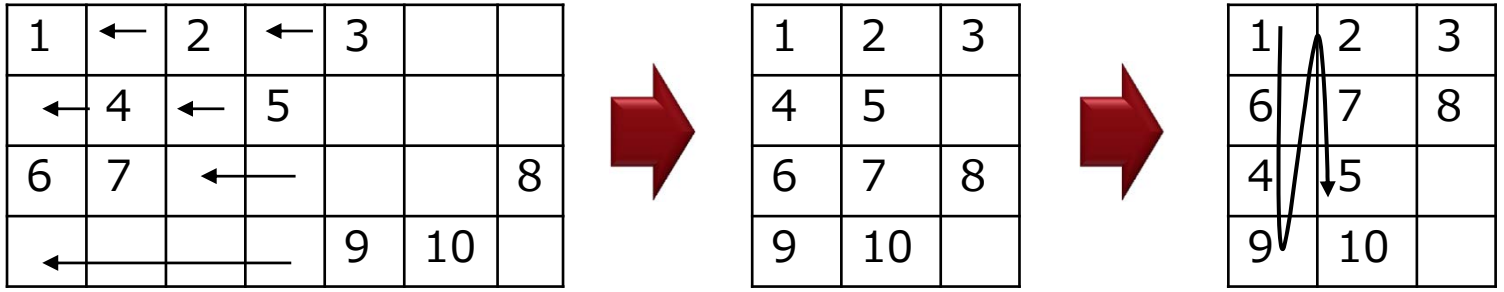


Vector length is limited by the number of non-zeros of each row

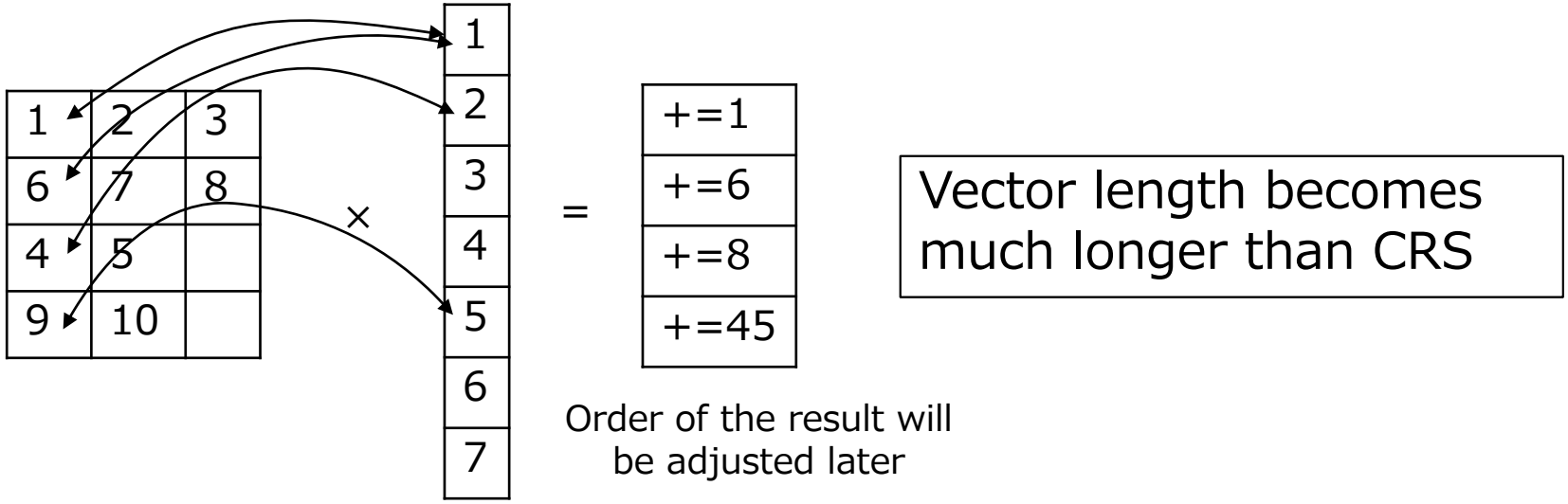
Optimization of SpMV for ML (3/5)

Existing matrix format (2): Jagged Diagonal Storage (JDS)

- shift the non-zeros to the left
- sort rows by the num. of non-zero; store them in col. major order



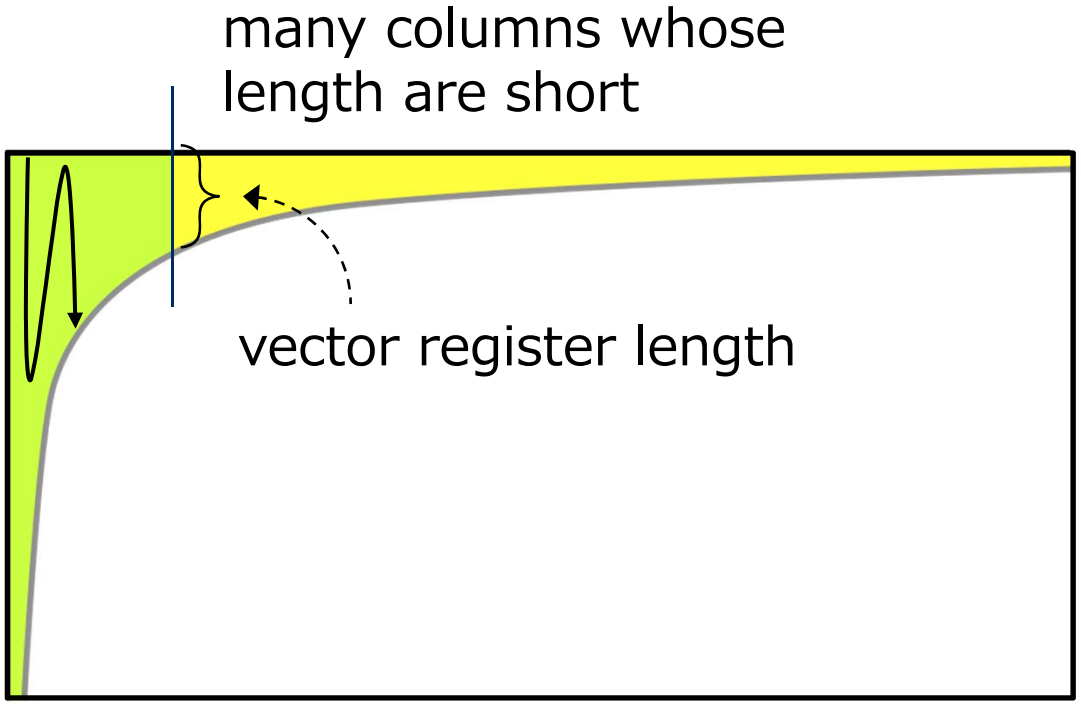
- SpMV is implemented by adding to the result vector



Optimization of SpMV for ML (4/5)

Problem of JDS

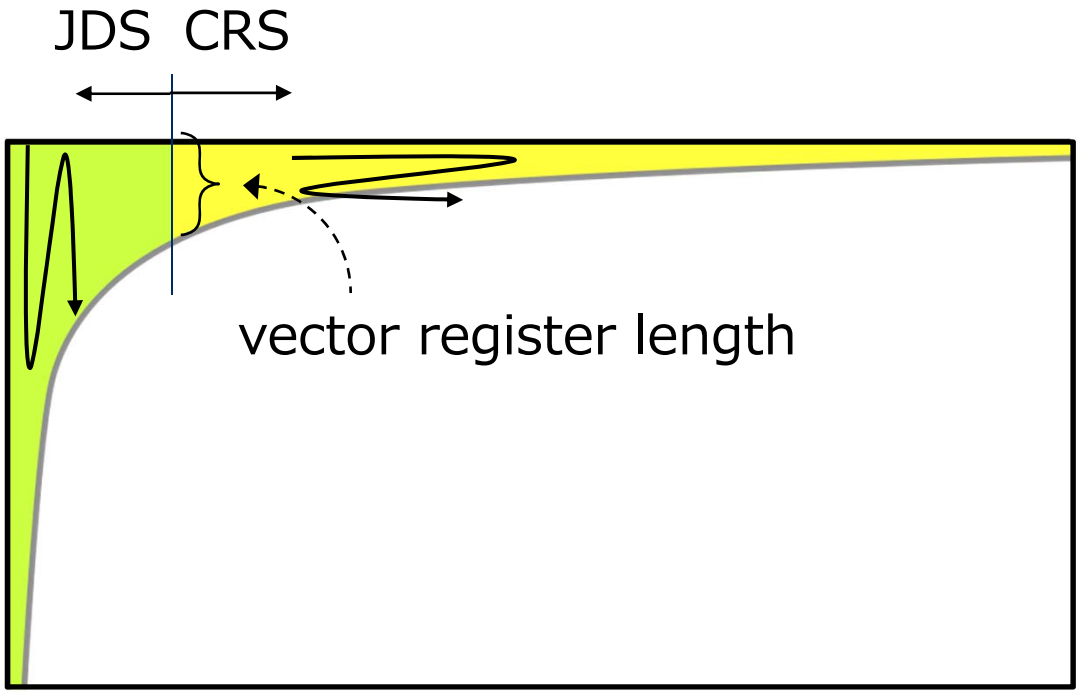
- If the distribution of non-zeros follows “power law”, many columns becomes shorter than vector register length



Optimization of SpMV for ML (5/5)

Solution: use the combination of both formats

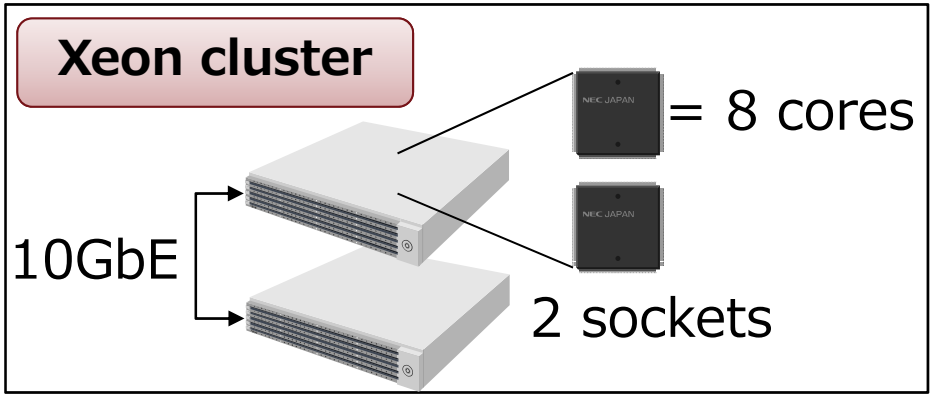
- use CRS format where column length are short

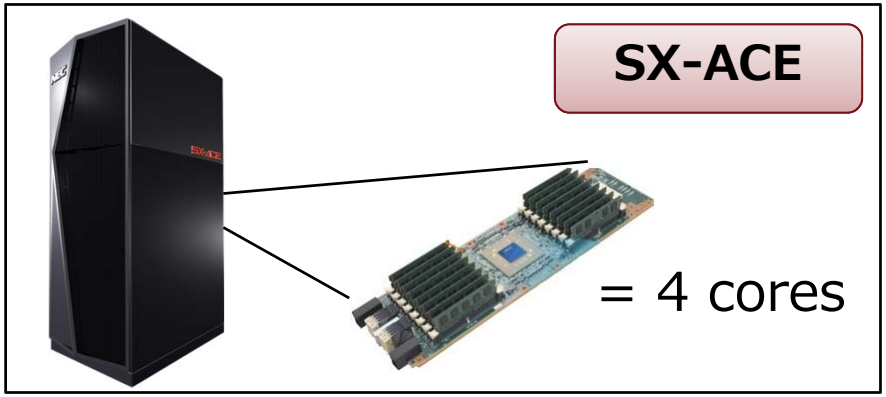


Evaluation: Evaluation Environment

Evaluated on Xeon and SX-ACE

- Xeon cluster (E5-2630v3)
  - 8 cores/socket, 2 sockets/node
  - 10Gbit Ethernet
  - Evaluated using Feliss and Spark
- SX-ACE
  - 4 cores/socket, 1 socket/node
  - 8GB/s network (bidirectional)
  - Evaluated using Feliss





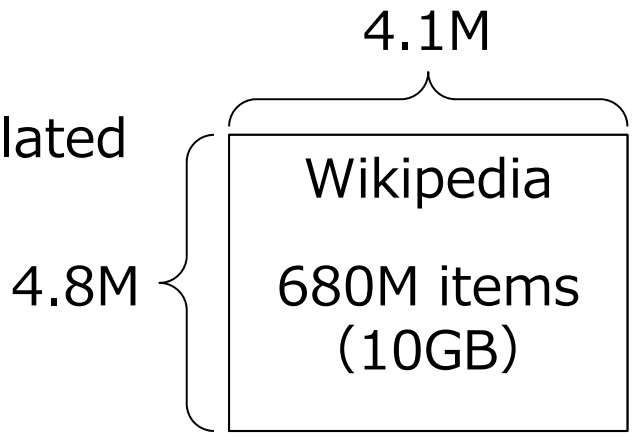
Evaluated only computation time

- without I/O time

Evaluation: Singular Value Decomposition (1/2)

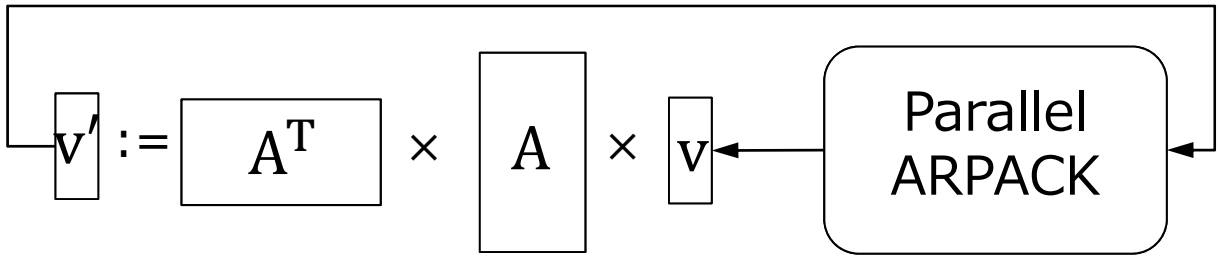
Evaluation setup

- Data: English Wikipedia document
- Top 100 singular values/vectors are calculated
- Corresponds to LSA



Implementation

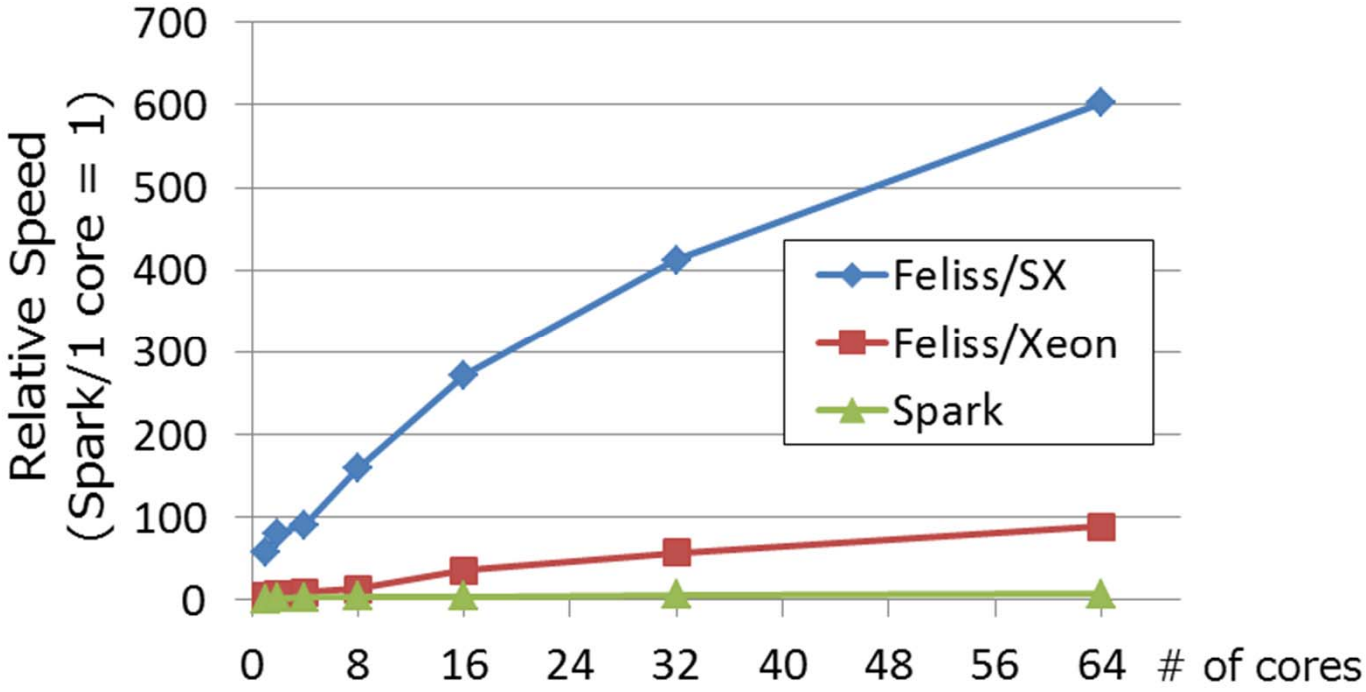
- Utilized Parallel ARPACK
  - requires user-defined SpMV
- Connected with our SpMV implementation
  - SpMV takes most of the time



Communication takes place
in both Parallel ARPACK and SpMV

Evaluation: Singular Value Decomposition (2/2)

- Compared at the same number of cores,
  - Vector on SX is 6.8 ~ 14x faster than Feliss on Xeon

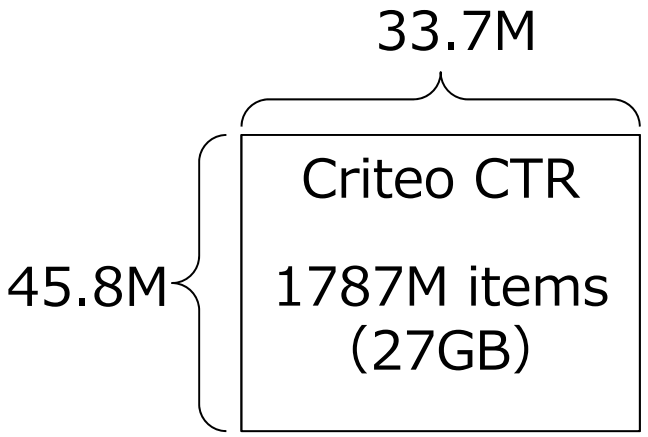


Feliss is our customized Spark for multi-processing

Evaluation: Logistic Regression (1/2)

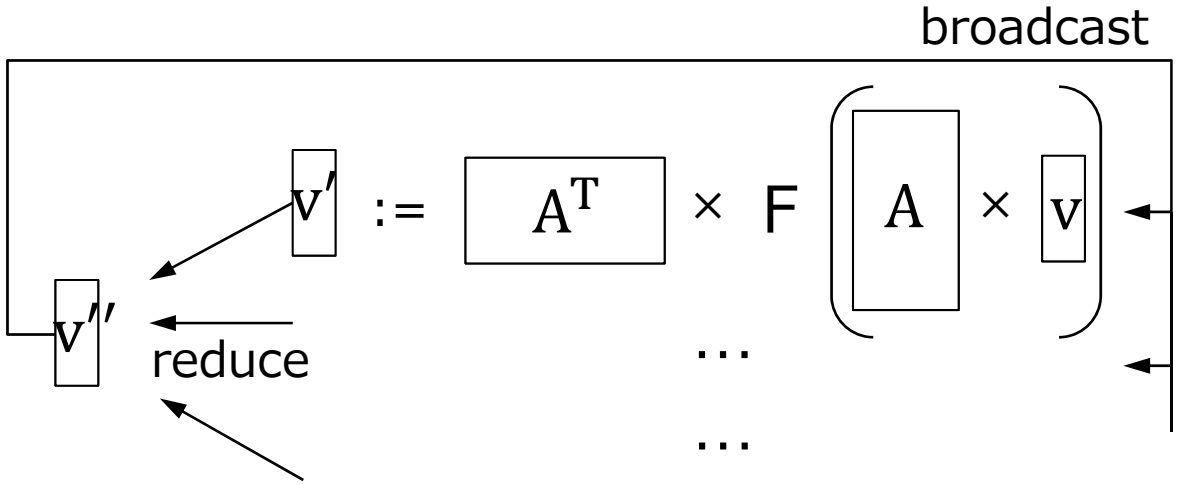
Evaluation setup

- Data: CTR data provided by Criteo
- Used “Gradient Descent” algorithm
  - Number of iteration = 100



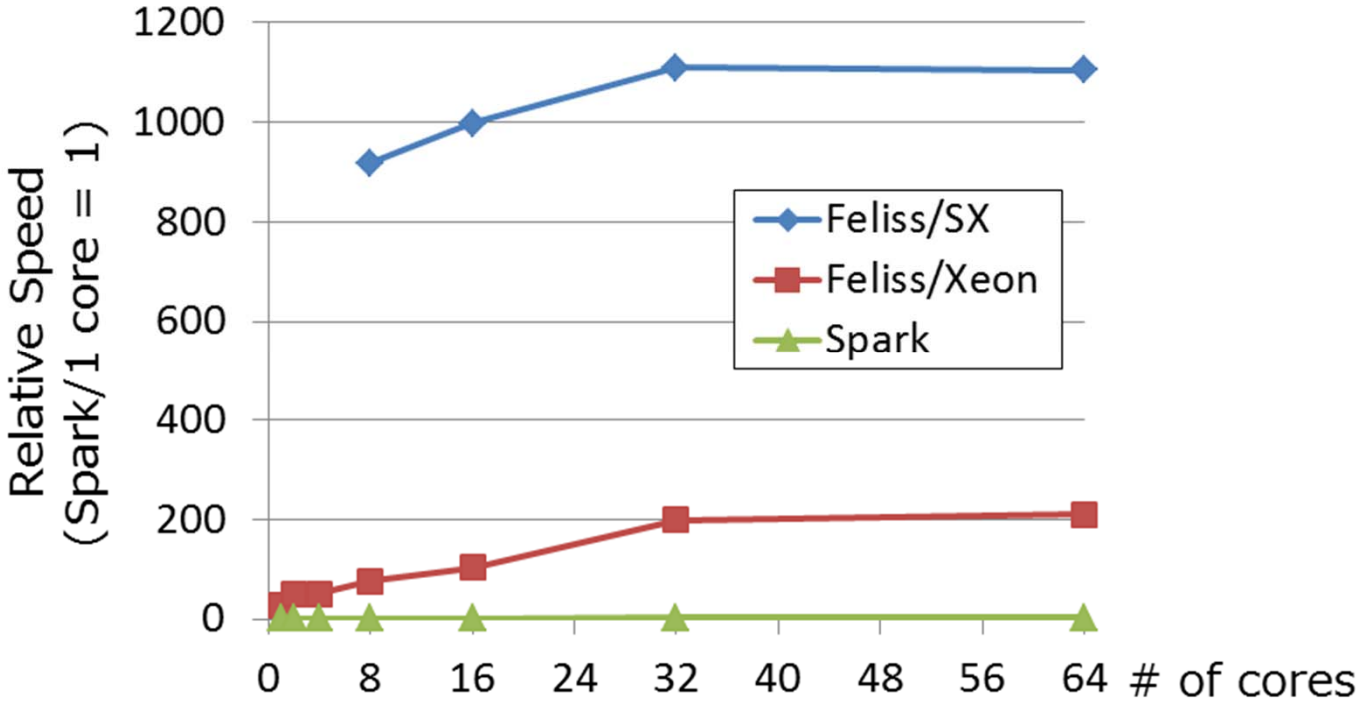
Implementation

- Divide and distribute the matrix
- Locally update the result using two SpMVs and a function (F)
- Reduce the result and broadcast



Evaluation: Logistic Regression (2/2)

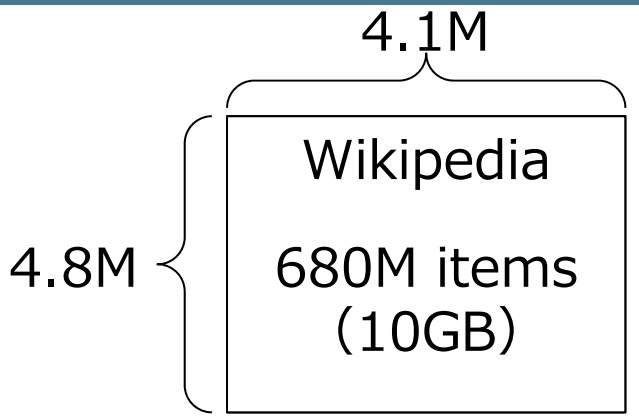
- Compared at the same number of cores,
  - Vector on SX is 5.2 ~ 12x faster than Feliss on Xeon
  - Scalability would be improved by using larger data



Evaluation: K-means (1/2)

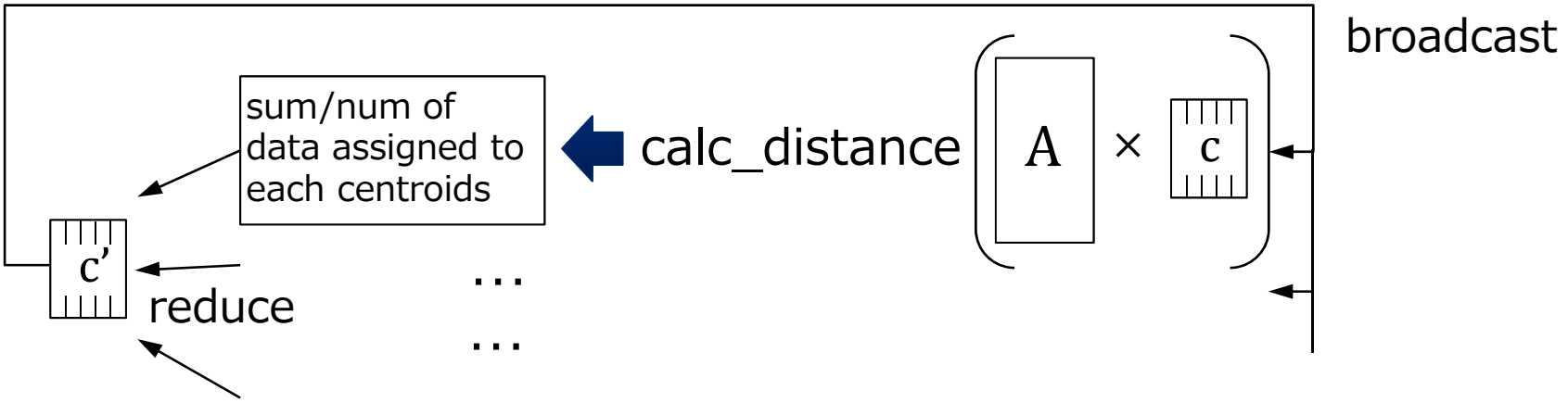
Evaluation setup

- Data: English Wikipedia document
- Number of clusters: 30
  - Number of iteration = 50



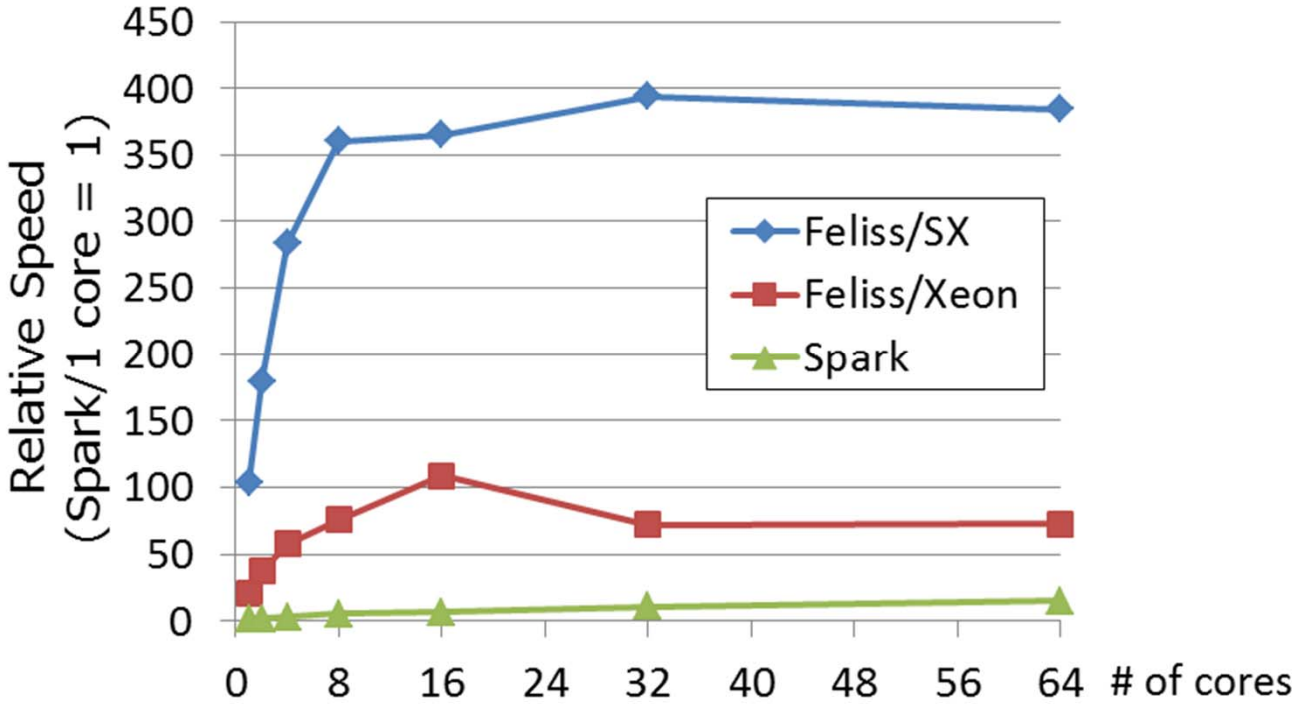
Implementation

- Divide and distribute the matrix
- Locally assign the data to the closest centroid (= center of cluster)
  - SpMV can be used to calc the distance
  - Distances to all centroids can be calculated at the same time (⇒ SpMM)
- Gathering the sum (and num) of the assigned data and create new centroids by averaging them



Evaluation: K-means (2/2)

- Compared at the same number of cores,
  - Feliss on SX is 3.4 ~ 5.4x faster than Feliss on Xeon
  - Scalability would be improved by using larger data



Conclusion

- To enhance computing speed under the limit of Moore' Law, Hetero computing is very important
- Vector processor is one of good accelerators in Hetero Computing
- We showed that users can easily write distributed programs on V without using MPI
- We showed effectiveness of a bigdata processing and Vector over Xeon using an ML algorithm


Orchestrating a brighter world
NEC