China Big Data and HPC Initiatives Overview

Xuanhua Shi

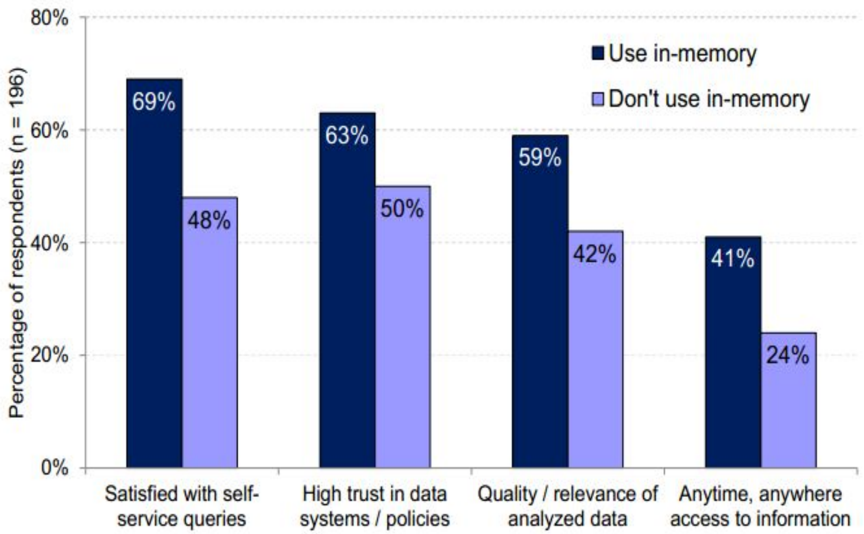
Services Computing Technology and System Laboratory
Big Data Technology and System Laboratory
Cluster and Grid Computing Laboratory
Huazhong University of Science and Technology, Wuhan, China
xhshi@hust.edu.cn

Outline

- In-Memory Computing
- New Funding by MOST
  - HPC Initiatives (2016-2020)
  - Big Data Initiatives (2016-2020)

In-Memory Computing: Lifting the Burden of Big Data – Aberdeen Group

Figure 3: Satisfaction and Trust in Business Data



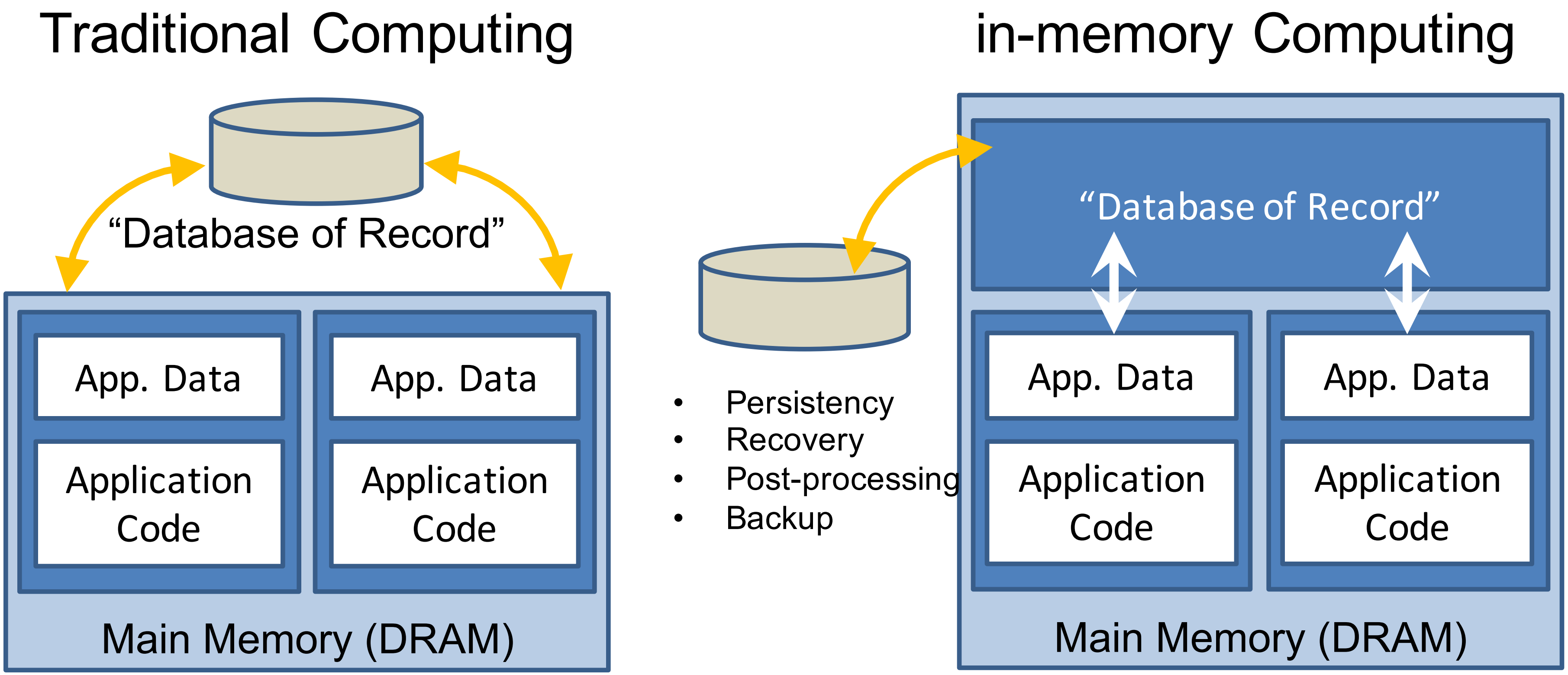
Source: Aberdeen Group, December 2011

Table 1: More Data, More Speed, More Efficiency

Performance Metrics	Use in-memory computing (n = 33)	Don't use (n = 163)	In-memory Benefit
Median amount of active business data	38 terabytes	18 terabytes	**2.1 times more data**
Median amount of data analyzed	14 terabytes (37% of all data)	4 terabytes (22% of all data)	**3.5 times more data**
Average response time for data analysis or query	42 seconds	75 minutes	**107 times faster**
Data volume processed per hour	1200 terabytes	3.2 terabytes	**375 times more efficient**

Source: Aberdeen Group, December 2011

In-memory Database

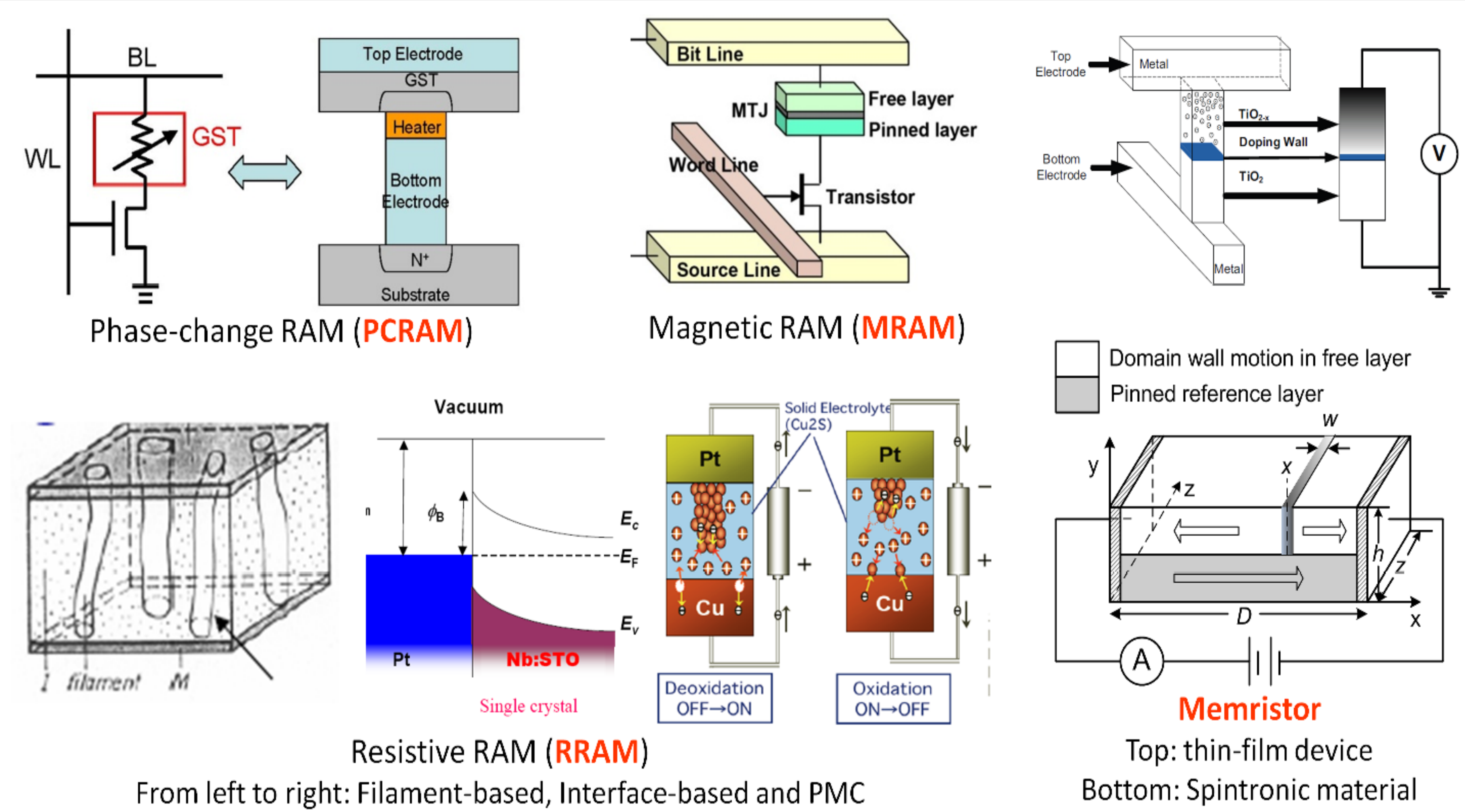


- Traditional systems exchange data pages between memory and disk when computing with big data : **Expensive IO cost**
- Speed of Disk: **ms** ≫ Speed of Memory: **ns**
- In-Memory Computing: CPU reads data from memory and provides real-time data processing.

Downsides of DRAM Refresh

- Energy consumption: Each refresh consumes energy
- Performance degradation: DRAM bank unavailable while refreshed
- QoS/predictability impact: (Long) pause times during refresh
- Refresh rate limits DRAM capacity scaling

Emerging Non-volatile Memory (NVM) Technologies

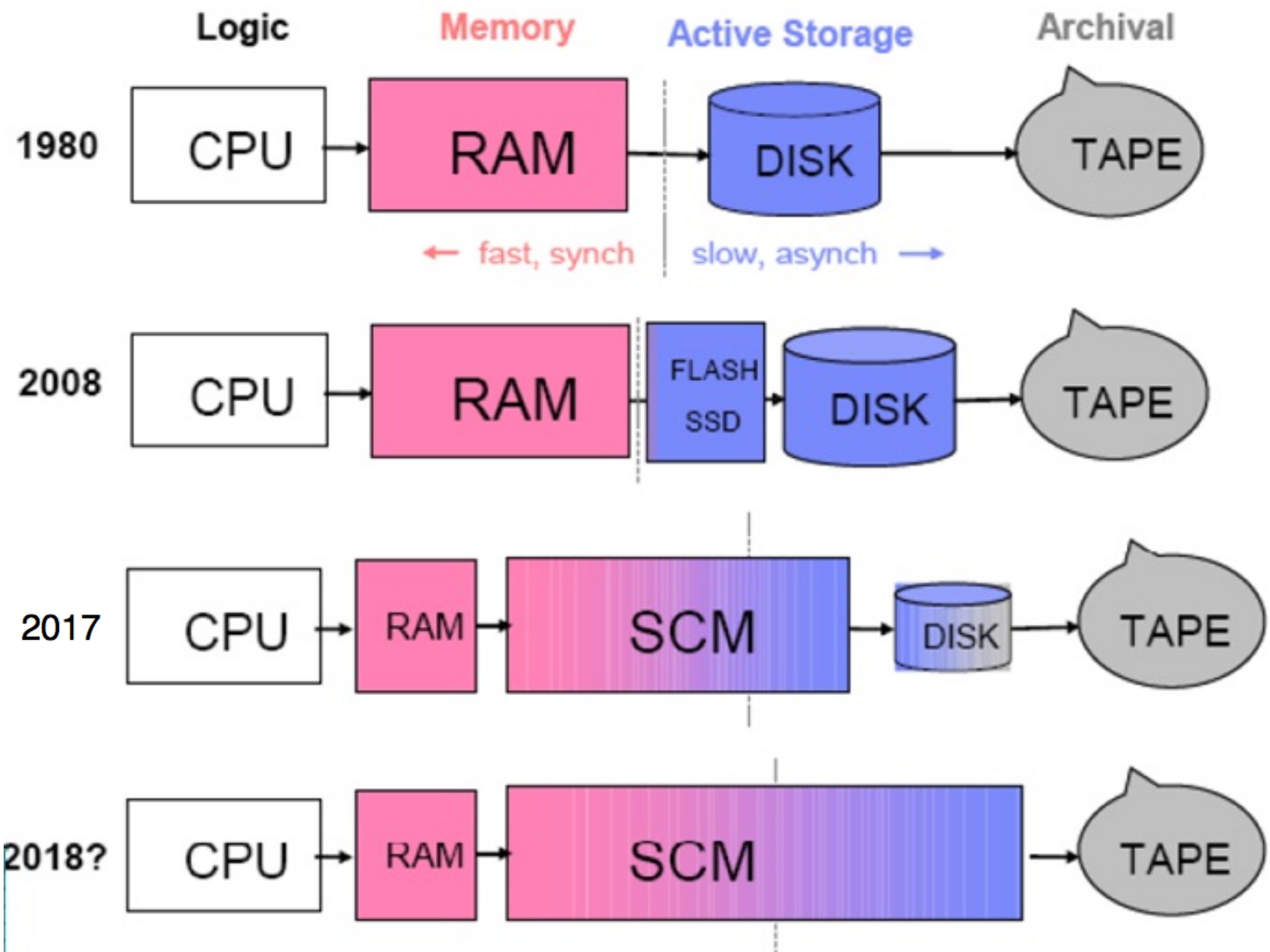


Emerging Non-volatile Memory (NVM) Technologies

Storage Class Memory

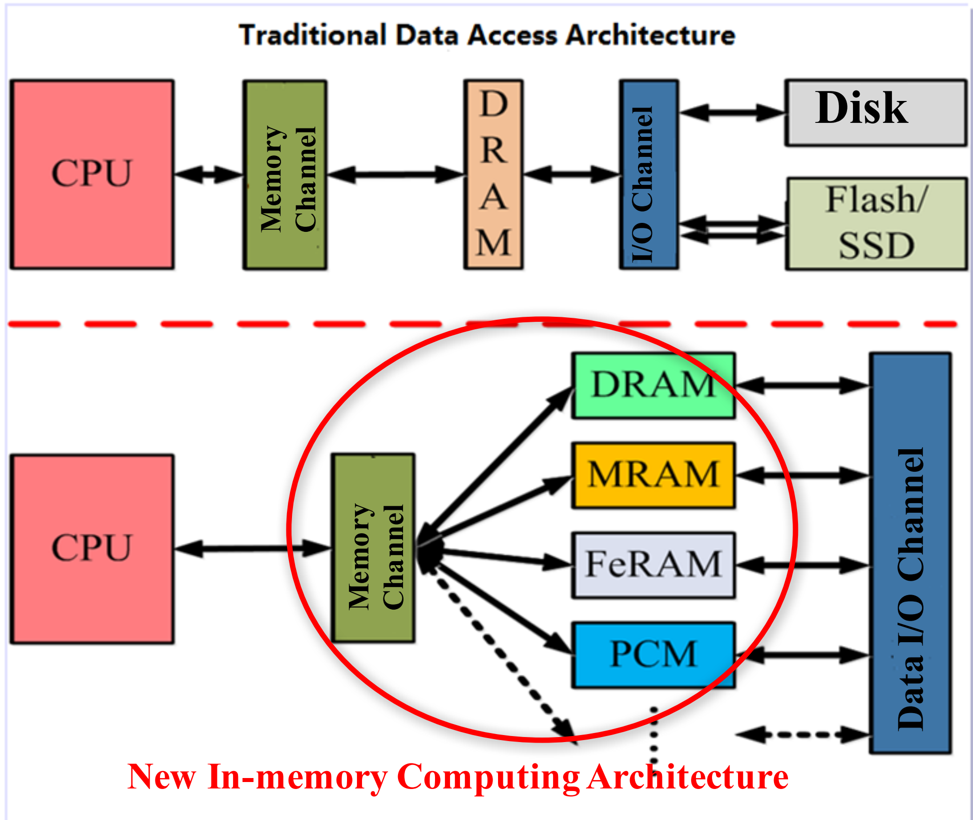
- SCM is a new class of data storage and memory devices
- SCM blurs the distinction between
  - MEMORY (= *fast, expensive, volatile*) and
  - STORAGE (= *slow, cheap, non-volatile*)
- Characteristics of SCM
  - Solid state, no moving parts
  - Short access times (~ DRAM like, within an order-of-magnitude)
  - Low cost per bit (DISK like, within an order-of-magnitude)
  - Non-volatile (~ 10 years)

Reconstruction of Virtual Memory Architecture: Break the I/O Bottleneck



New In-Memory Computing Architecture

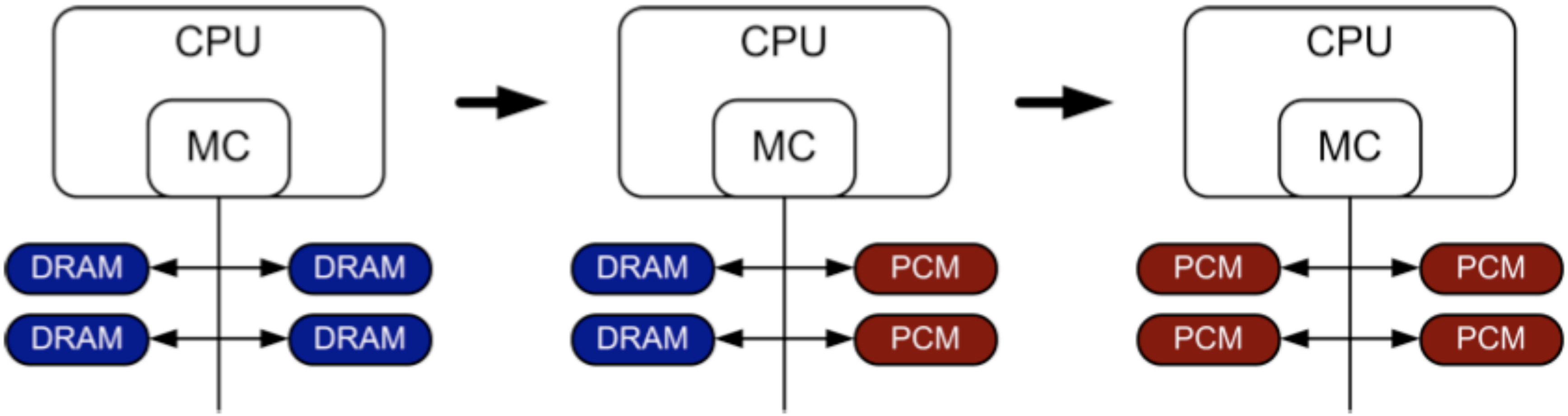
Build DRAM + SCM hybrid hierarchical/parallel memory structure



The new in-memory computing architecture supports the shift from computing-centric to combination of computing and data

Challenges of Hybrid Memory Systems

- How should SCM-based (main) memory be organized?



- Partitioning
  - Should DRAM be a cache or main memory, or configurable?
  - What fraction? How many controllers?

Challenges of Hybrid Memory Systems

- Data allocation/movement (energy, performance, lifetime)
  - Who manages allocation/movement?
  - What are good control algorithms?
  - How do we prevent degradation of service due to wearout?
- Design of cache hierarchy, memory controllers, OS
  - Mitigate PCM shortcomings, exploit PCM advantages
- *Persistent* data can be *randomly* and *synchronously* addressed
  - Huge non-volatile address spaces, memory-mapped DB, persistent objects…
  - Should SCM be used like I/O or like memory or in a totally new way?

Challenges of Hybrid Memory Systems

- Software Architecture
  - *S*hould one make SCM *visible* to applications software?
  - If visible, in which form?
    - New APIs, libraries, memory models, new I/O devices,...
- Databases, Business Intelligence and Streams are first impacted
  - Data-intensive HPC - predictable execution time of complex business analytics - streaming search

Challenges of Enabling and Exploiting NVM

- Enabling NVM and hybrid memory
  - How to tolerate errors?
  - How to enable secure operation?
  - How to tolerate performance and power shortcomings?
  - How to minimize cost?
- Exploiting emerging technologies
  - How to exploit non-volatility?
  - How to minimize energy consumption?
  - How to exploit NVM on chip?

Technology and System of In-Memory Computing for Big Data Processing

- Hybrid Memory Architecture for In-Memory Computing System
- System Software for In-Memory Computing System
- Parallel Processing for In-Memory Computing System
- Data Management for In-Memory Computing System

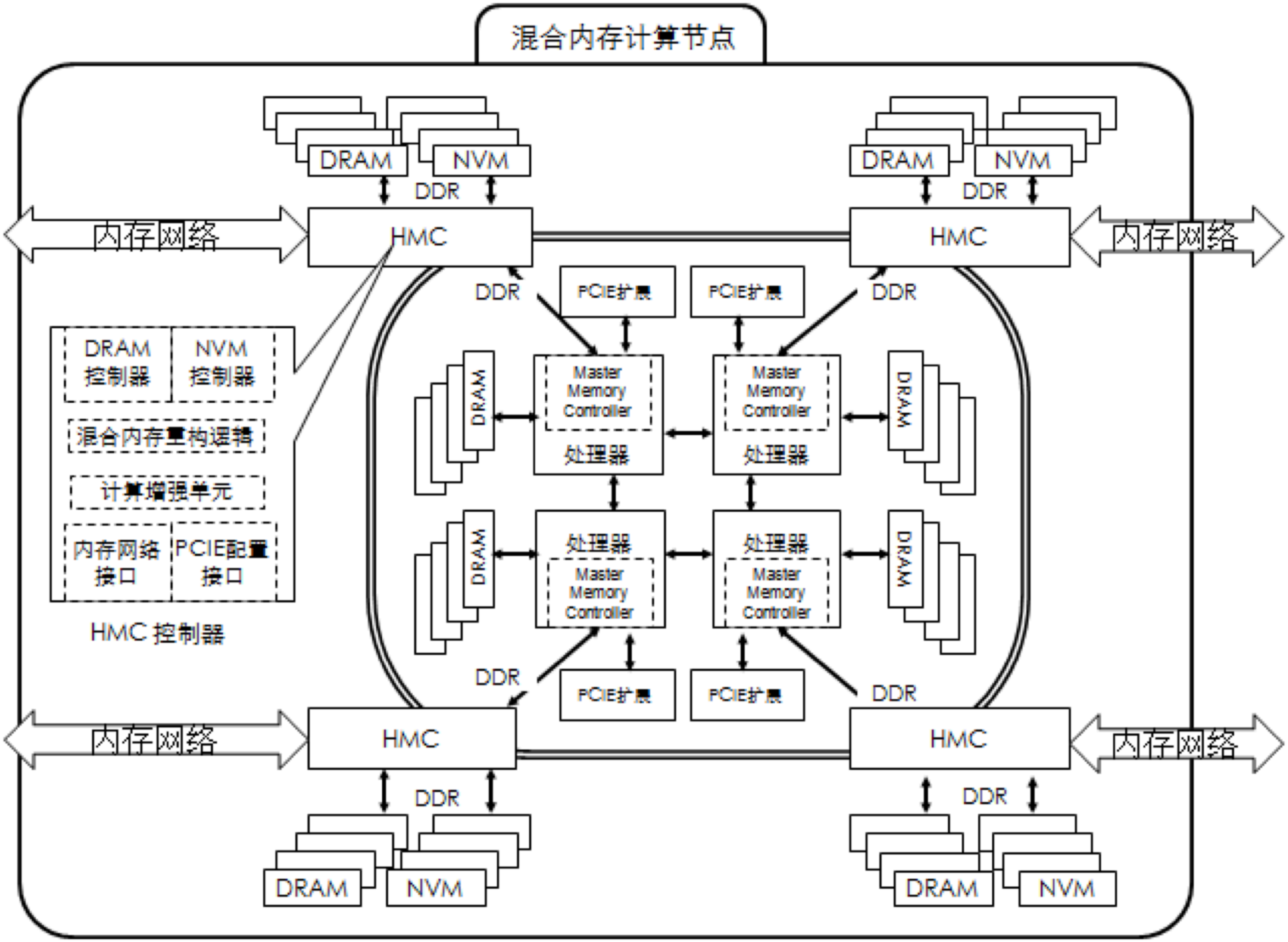
Technology and System of In-Memory Computing for Big Data Processing

- Project Overview
  - Total budget: 170M RMB
  - Period: January 2015 – December 2017
  - Participants
    - Inspur
    - Huawei
    - Sugon
    - Shanghai Jiaotong University
    - Huazhong University of Science and Technology
    - Chongqing University
    - National University of Defense Technology
    - Huadong Normal University
    - Jiangnan Computing Institute

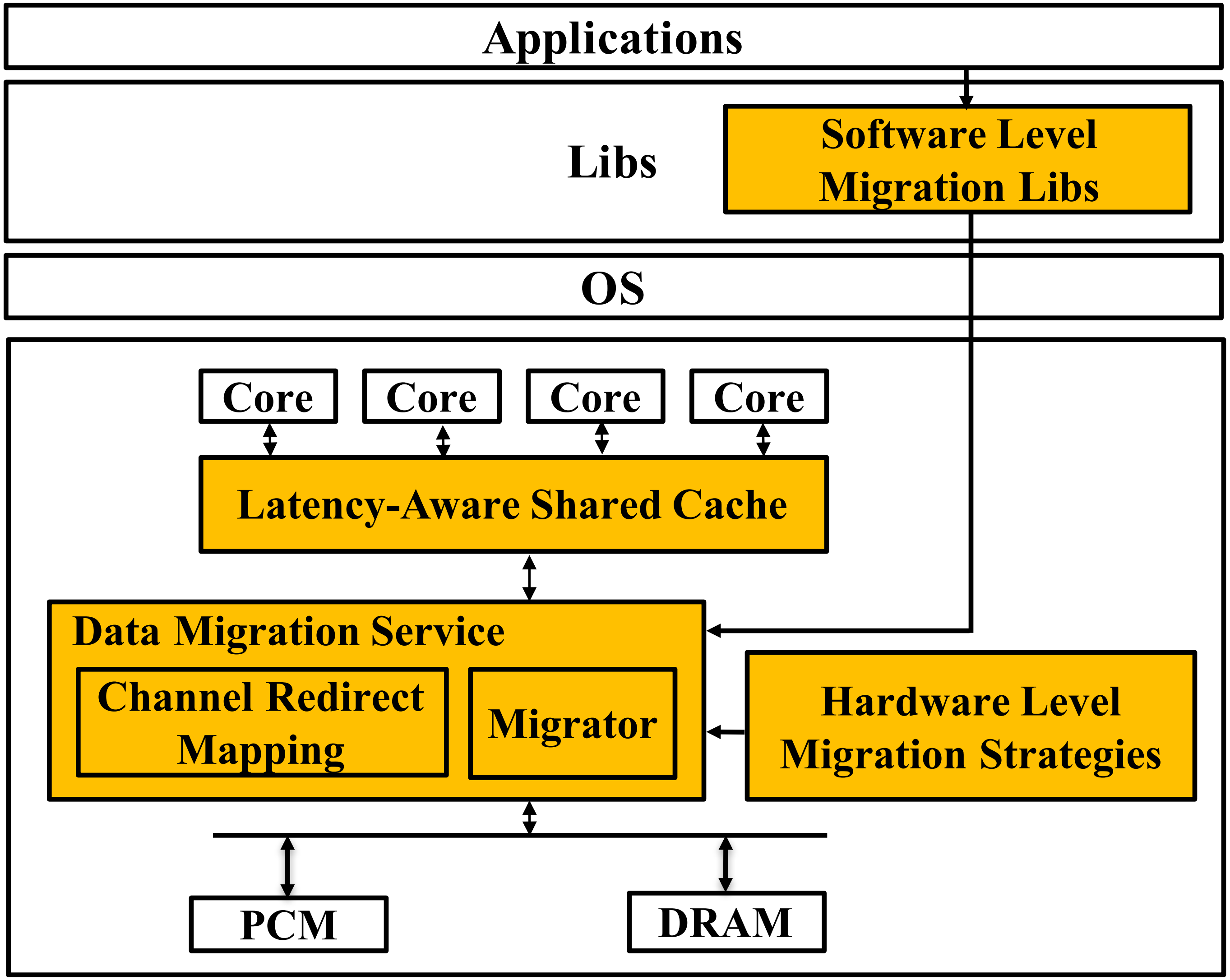
Technology and System of In-Memory Computing for Big Data Processing

- Project Mission
  - Hybrid NVM-based high reliable, massive storage, and low power in-memory computing system, the capacity of NVM in each node should be in TB level, supporting zero bootup
  - System software and simulation platform for in-memory computing system
  - Parallel processing system for in-memory computing system
  - In-memory database for hybrid memory architecture to support decision making and other data management applications

System Architecture



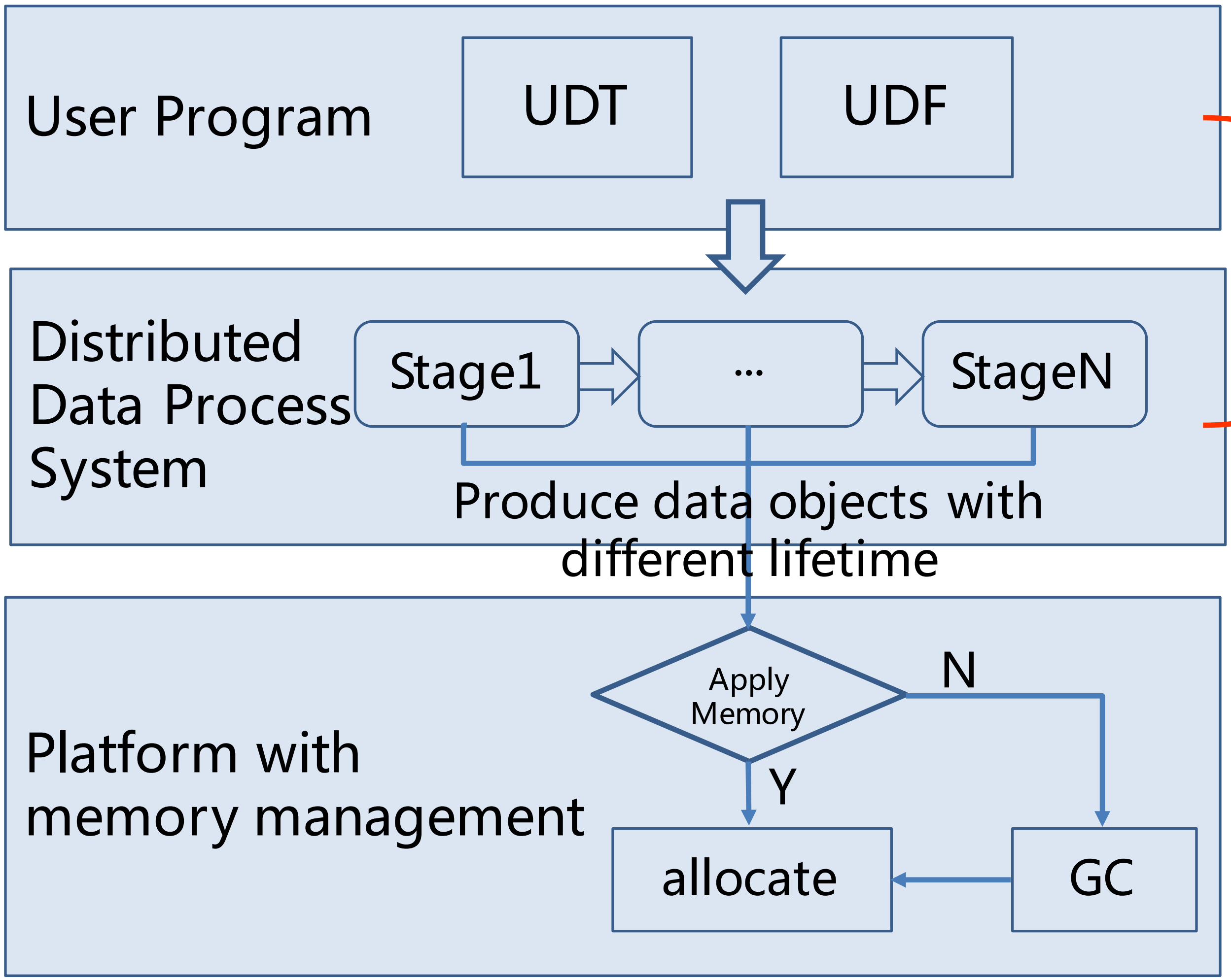
Full System Simulator



Designed Based On:
MARSSX86+
NVMain

- More flexible (supports NVMain、DRAMSim、HybridSim as main memory simulator)
- Simulate memory system precisely, easy to configure

Pros and Cons of DDPS

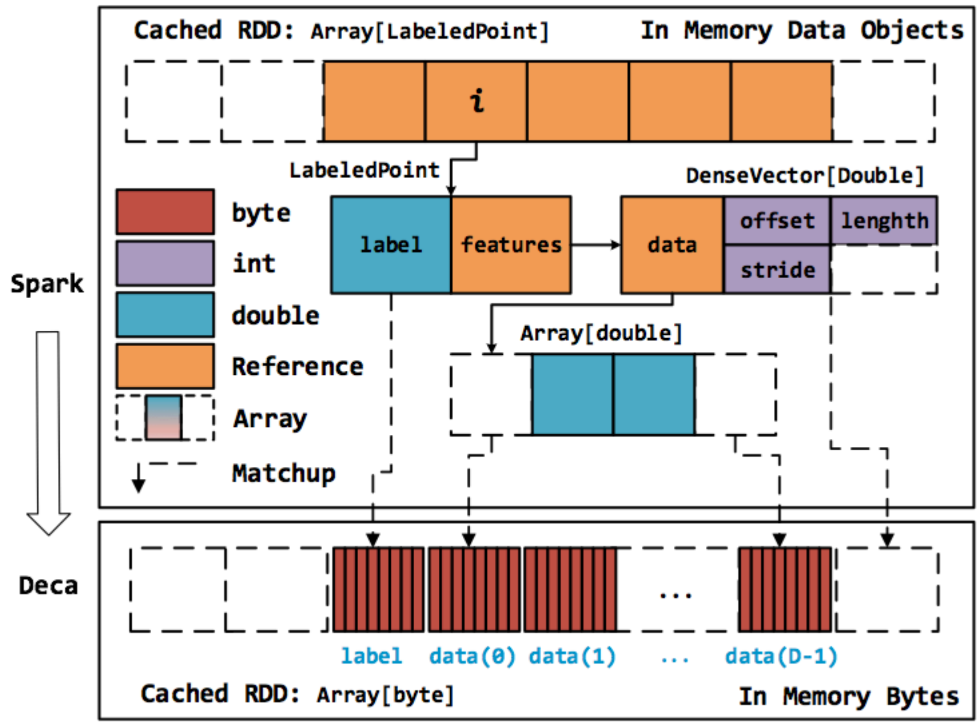


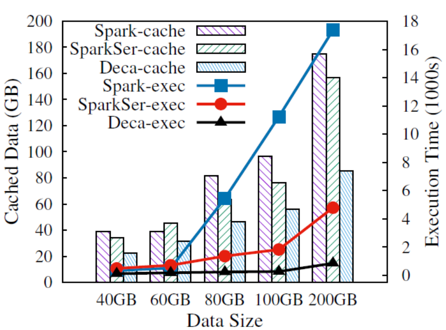
Benefit from the high-level object-oriented language

The automatic memory management of platform effect the performance seriously。

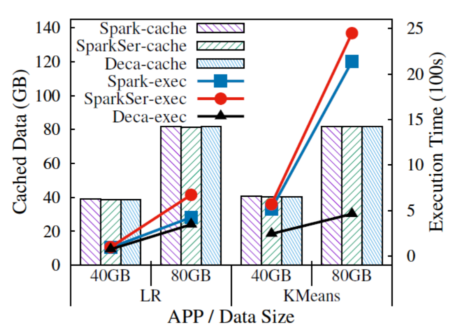
Deca: Exploiting Raw Data of In-Memory Data Objects in Distributed Data-Parallel Systems

- Completely decomposing in-memory data objects to eliminate the references invocation and reduce frequent garbage collection in JVM
- A system to automatic converse the user codes, decompose data objects and manage in-memory raw data
- Speedup from 22.7x to 41.6x, compare with Apache Spark





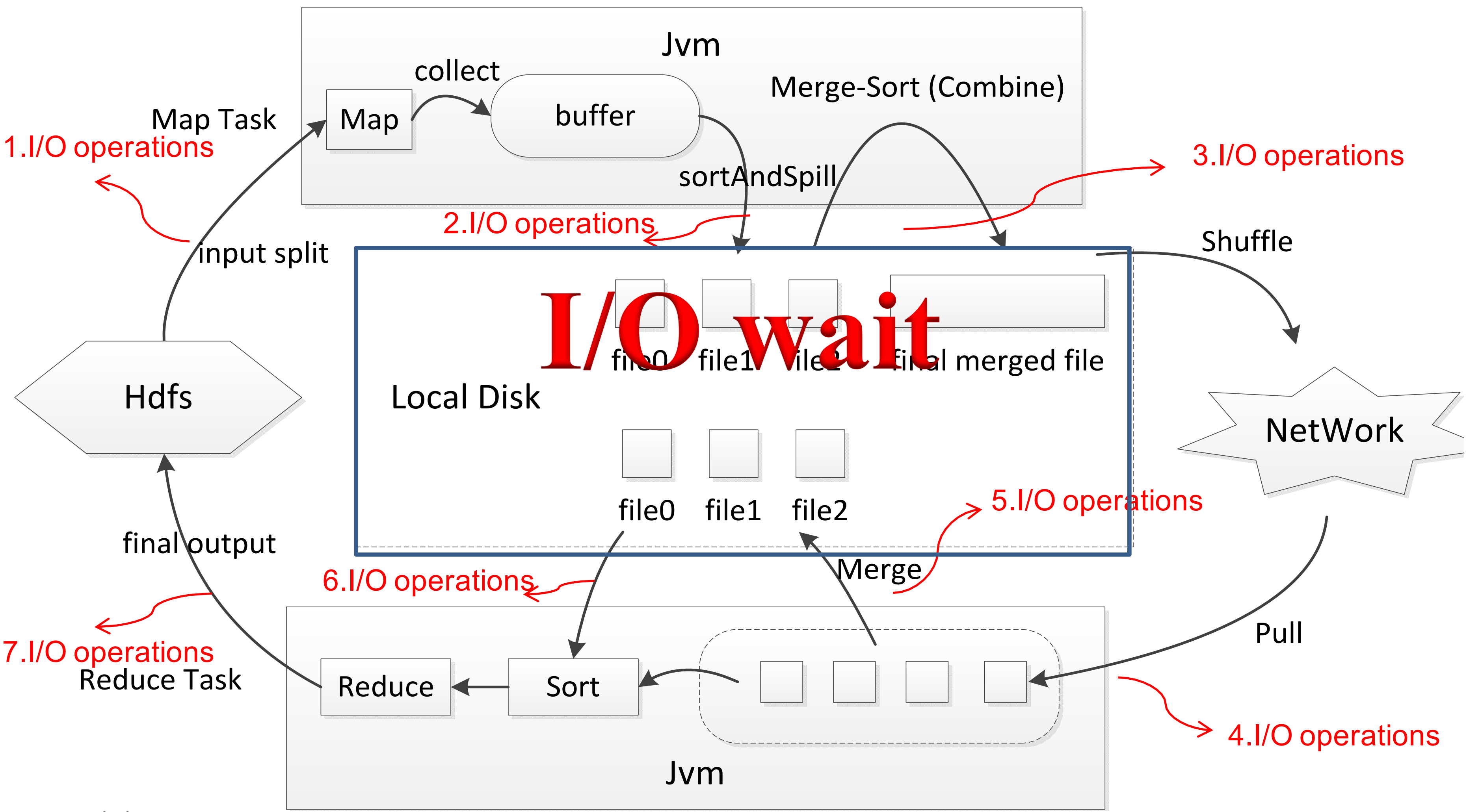
(c) KMeans



(d) Amazon Image Dataset

App	Spark			Deca	
	exec.	gc	ratio	gc	reduction
WC: 150GB	4980s	2016s	40.5%	12.2s	99.4%
LR: 80GB	2820s	2069.9s	73.4%	2.5s	99.9%
KMeans: 80GB	5443s	4294.8s	78.9%	7.2s	99.8%
PR: 30GB	5544s	3588.6s	64.7%	21.7s	99.4%
CC: 30GB	2088s	1443.9s	69.2%	36s	97.5%

Hadoop Engine: IO-intensive



Mammoth: Memory-Centric MapReduce System

- A novel rule-based heuristic to prioritize memory allocation and revocation mechanism
- A multi-threaded execution engine, which realize global memory management
- Compatible with Hadoop
- Sources available at Github and ASF
- IEEE Computer Spotlight

SPOTLIGHT ON TRANSACTIONS

When Data Grows Big

Hai Jin, Huazhong University of Science and Technology

This installment of *Computer*'s series highlighting the work published in IEEE Computer Society journals comes from *IEEE Transactions on Parallel and Distributed Systems*.

Hadoop is an open source software framework that uses the well-known MapReduce model to process large-scale datasets. It's widely used by many data processing companies including Google, Yahoo, Facebook, and LinkedIn. Most of these have dedicated Hadoop clusters, which have abundant memory to achieve high system throughput. However, many smaller companies, research institutes, and universities might only have access to high-performance computing (HPC) or ordinary commodity clusters, which are both memory-constrained compared to Hadoop.

The latest survey conducted by the International Data Corporation (IDC) indicates that 67 percent of HPC systems are now used for big data analysis. It's unclear whether the MapReduce model can reach its full potential in these constrained platforms. If it can't, how might we re-engineer the traditional Hadoop system toward this purpose?

In the forthcoming article "Mammoth: Gearing Hadoop towards Memory-Intensive MapReduce Applications" (*IEEE Transactions on Parallel and Distributed Systems*; DOI 10.1109/TPDS.2014.2345068), the authors conducted benchmarking experiments with Hadoop and observed inefficiencies in both memory usage and I/O operations. These deficiencies cause significant performance reduction in Hadoop, especially when the supporting platform's memory is constrained.

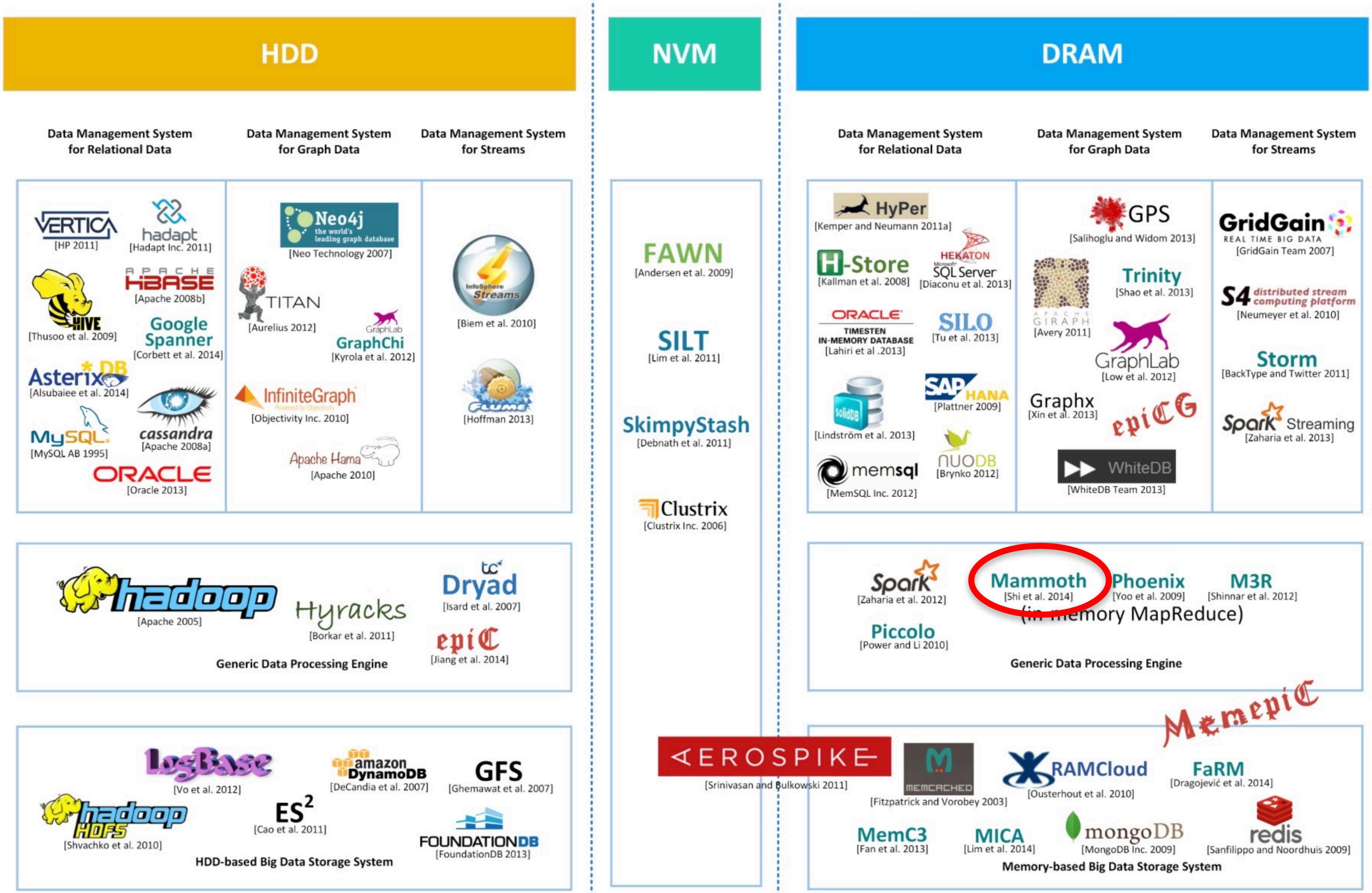
The authors observed static and coarse-grained memory management inefficiencies in Map and Reduce tasks; unnecessary disk spilling during the Map/Reduce procedure; lack of coordination among the Map tasks with different memory demands; excessive I/O waits caused by the merge-sort procedure; excessive disk seeks caused by the parallel I/O; and the long-tail effect caused by an inappropriate priority setting for the file buffer. To tackle these problems, the authors developed a new MapReduce data processing system called Mammoth for memory-constrained systems.

Mammoth is a multi-thread execution engine that's based on Hadoop but runs in a single JVM on each node. Each Map or Reduce task on a node is executed as a thread in the engine, and all task threads can share memory at runtime. A memory-scheduling algorithm is developed in the execution engine to realize global memory management. The authors further implemented the techniques of disk access serialization, multi-cache, and shuffling from memory, and also solved the problem of full garbage collection in the JVM. The authors also designed a novel rule-based heuristic to prioritize memory allocation and revocation among execution units (mapper, shuffler, reducer, and so on), which maximizes the holistic benefits of the Map/Reduce job when scheduling each memory unit.

The authors conducted extensive experiments to compare Mammoth with Hadoop and another popular in-memory processing framework called Spark. The results show that Mammoth can dramatically improve performance in terms of execution time on memory-constrained clusters.

Hai Jin is a Cheung Kung Scholars Chair professor of computer science and engineering at Huazhong University of Science and Technology (HUST) and dean of the School of Computer Science and Technology at HUST. Contact him at hjin@hust.edu.cn.

Landscape of Disk-Based and In-Memory Data Management Systems (2014)



Outline

- In-Memory Computing
- New Funding by MOST
  - HPC Initiatives (2016-2020)
  - Big Data Initiatives (2016-2020)

HPC Initiatives (2016-2020)

- Two 100PFlops Supercomputer
  - One is located Wuxi, another is located in Guangzhou
- E-scale architecture
- E-scale processors
- High-speed network
- HPC software stack
- Co-design: Aircraft design and Weather forecasting
- Some typical applications

Big Data Initiatives (2016-2020)

- Big data infrastructure
  - New storage system
  - Data flow based data analyzed stack
  - Domain specific data management system
- Data-driven software
- Data analyze applications and Human-like intelligence
  - From data to knowledge
  - Large scale objects recognition

Thanks!