课程所覆盖的专题

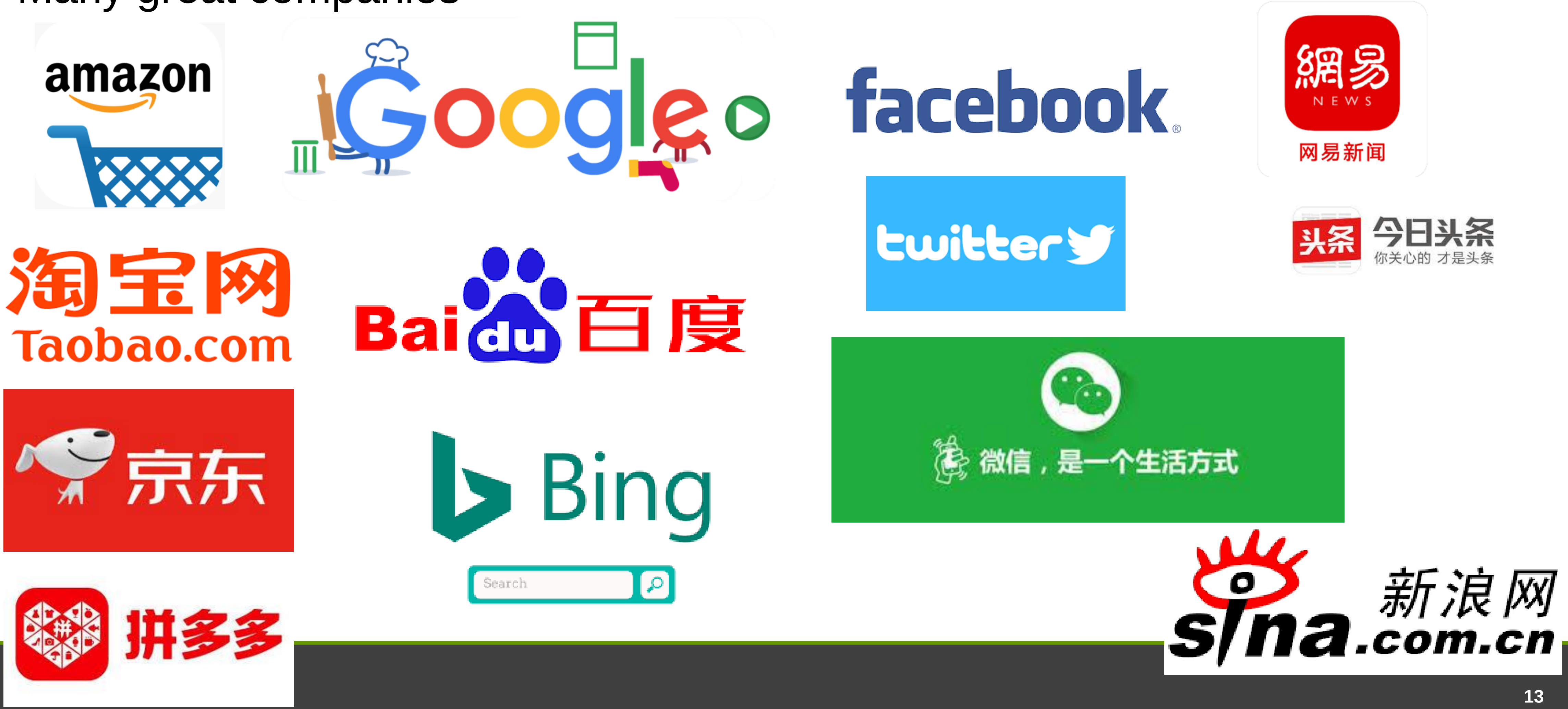
1. **简介**
2. **商务思维 (Business thinking)**
   - 所谓的“商务 (BUSINESS)” – 其实就是学会做出获得更多利润的决策 (making decisions to earn more profit)
   - 管理技巧 (Management skills) – 如何落实那些决策
   - 试试创业？ – 可以！但是要慎重！！
3. **数据分析的方法概览 (Data Analytics methods)**
   - 其实，数据分析有着悠久的历史 (HISTORY view about Data Analytics)
   - 理解数据分析方法的 – 一点优化的技巧 (OPTIMIZATION)
   - 来自统计学的数据分析方法 (STATISTICS) – 基于抽样的推断
   - 来自机器学习的数据分析方法 (BASIC + ADVANCED) – 基于数据的知识发现
4. **实用技巧 (Practical skills)**
   - 大商务，需要大数据
   - 大商务的两个挑战: “秒杀” 和 “精准广告”
5. **课程总结**

大商务，需要大数据

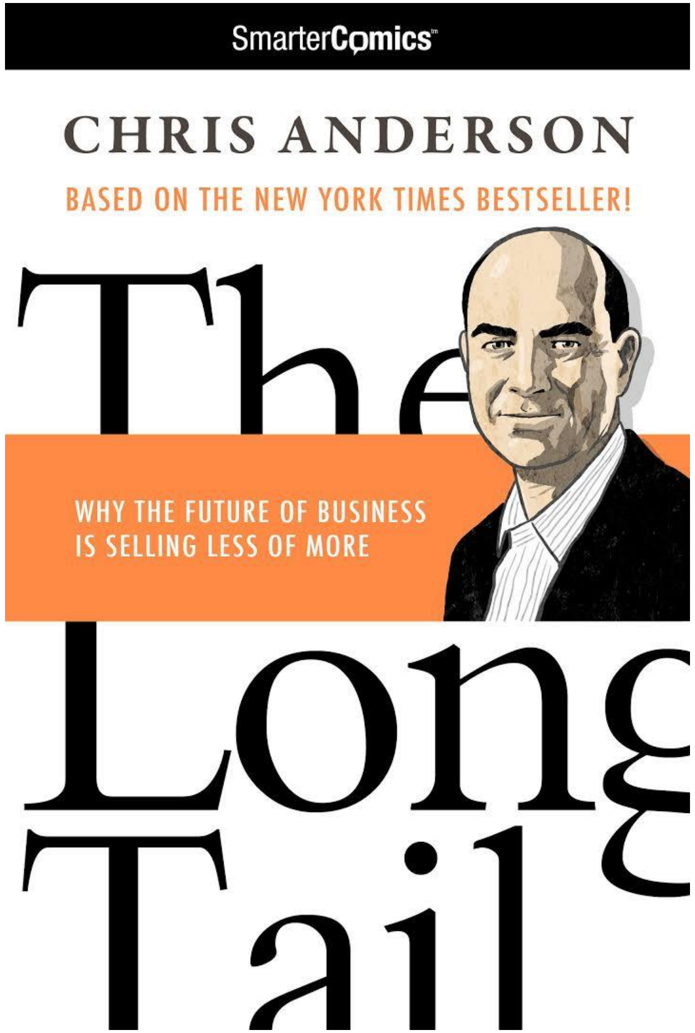
- **In IT age, <u>platform sticking consumers</u> [黏着客户的平台] is the popular pattern for e-business – Amazon, Google, Alibaba, JD, …**
  - **Large scale** (data and computing power) is important, which needs HPC
- **Large Scale Data – Big Data**
  - From File to Big Data
- **Large Scale computing power – High Performance Computing is now popular for business**
  - According to Top 500, MPP and Cluster
- **Additional bonus for Scientific Computing**
  - Weather forecasting

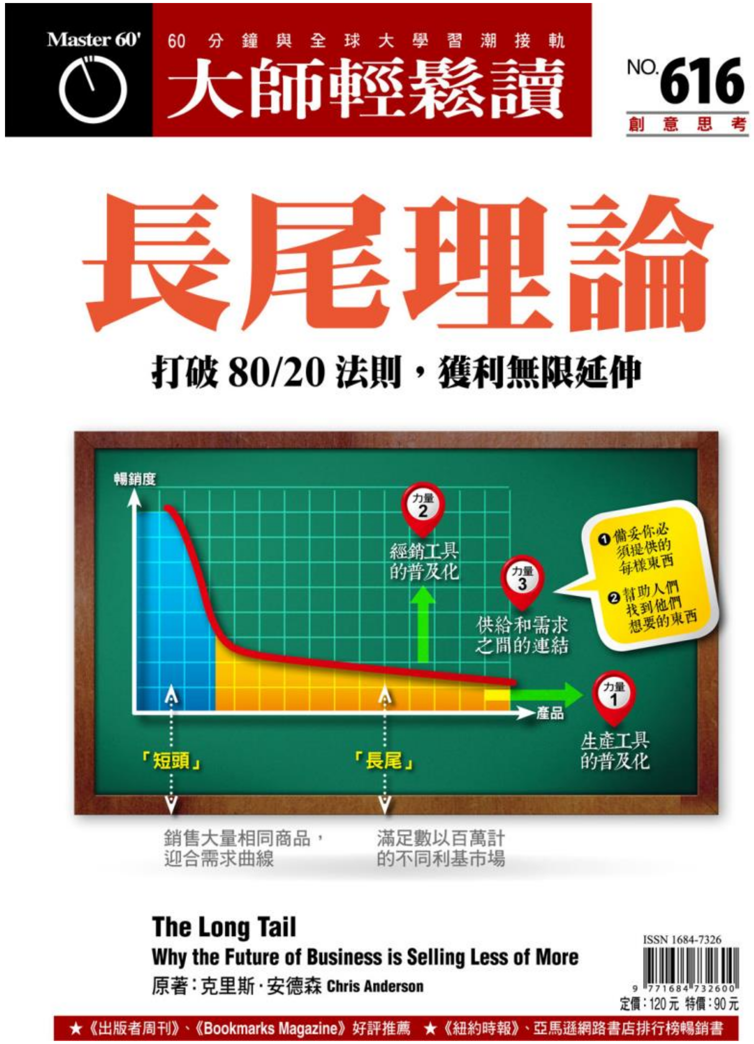
We are now in IT age/ML age/AI age …

- Platform is a popular business pattern for e-commerce
  - Many great companies

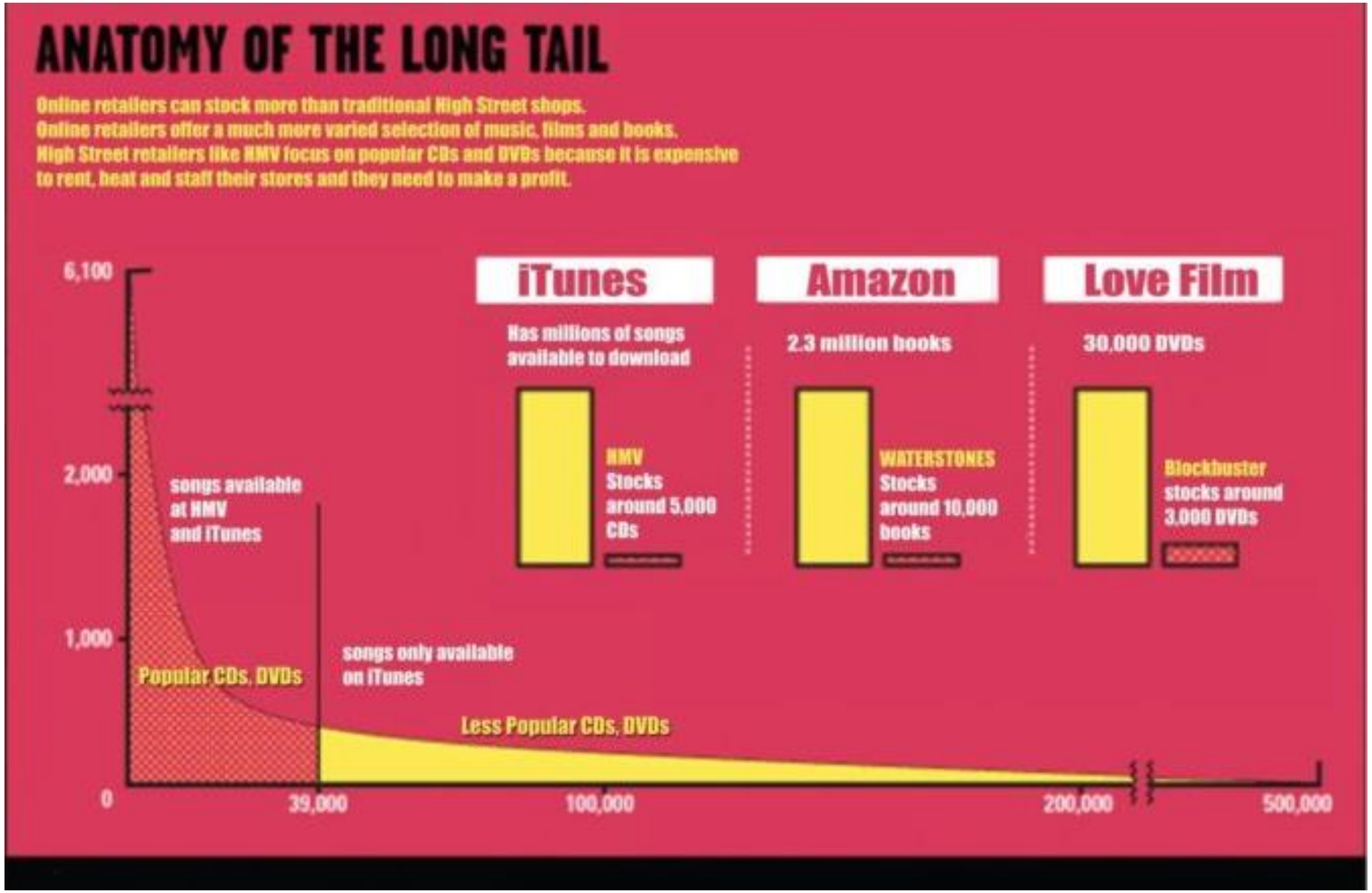


Long tail explains the Business @ IT age/E-commerce era


SmarterComics™
CHRIS ANDERSON
BASED ON THE NEW YORK TIMES BESTSELLER!
The Long Tail
WHY THE FUTURE OF BUSINESS
IS SELLING LESS OF MORE


Master 60'
60 分 鐘 與 全 球 大 學 習 潮 接 軌
大師輕鬆讀
NO.616
創 意 思 考
長尾理論
打破 80/20 法則，獲利無限延伸
暢銷度
力量 2
經銷工具
的普及化
力量 3
供給和需求
之間的連結
❶ 備妥你必須提供的每樣東西
❷ 幫助人們找到他們想要的東西
產品
力量 1
生產工具
的普及化
「短頭」
「長尾」
銷售大量相同商品，
迎合需求曲線
滿足數以百萬計
的不同利基市場
The Long Tail
Why the Future of Business is Selling Less of More
原著：克里斯‧安德森 Chris Anderson
ISSN 1684-7326
9 771684 732600
定價：120 元 特價：90 元
★《出版者周刊》、《Bookmarks Magazine》好評推薦 ★《紐約時報》、亞馬遜網路書店排行榜暢銷書

They are similar with the popular properties



ANATOMY OF THE LONG TAIL

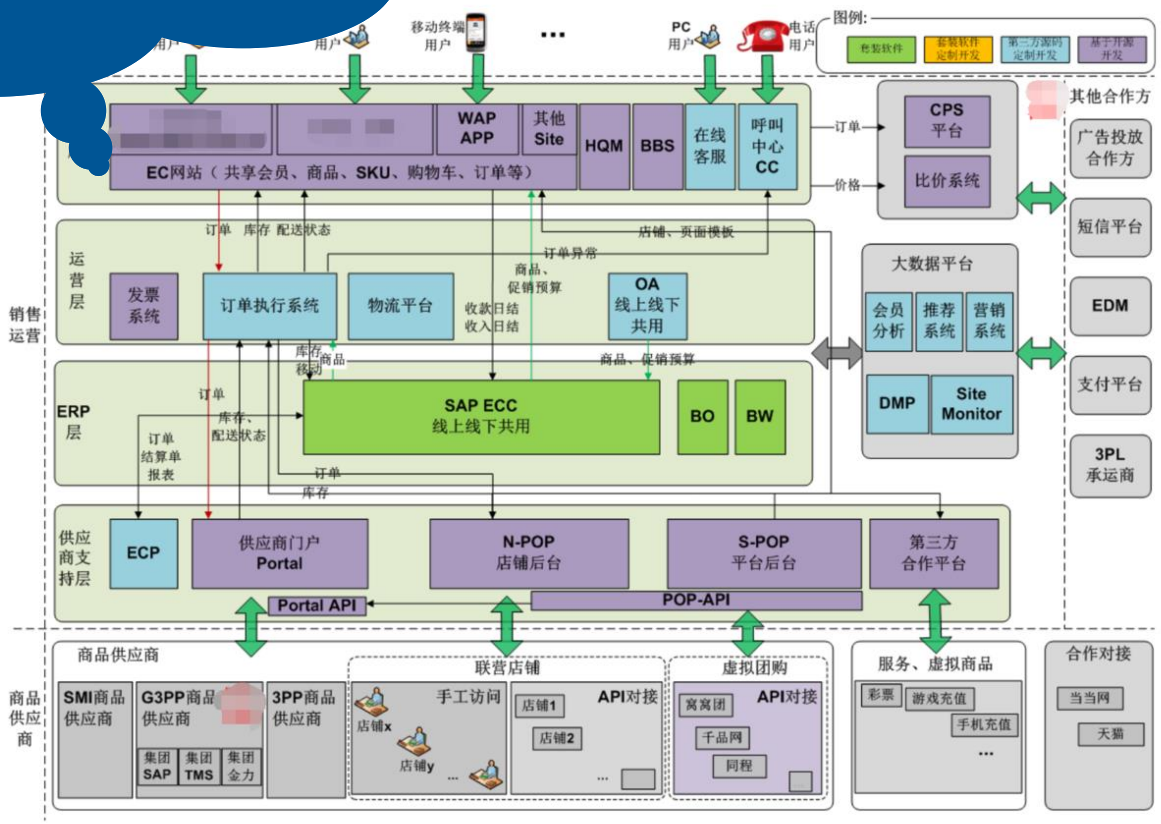
Online retailers can stock more than traditional High Street shops.
Online retailers offer a much more varied selection of music, films and books.
High Street retailers like HMV focus on popular CDs and DVDs because it is expensive to rent, heat and staff their stores and they need to make a profit.

- With **Website**
  - **Unlimited** rack [货架], products
    - Any product always has its own consumer
    - Search engine
  - Low storage cost, logistic cost
  - **Recommender system, Computing Advertising [计算广告]**

For me

Computers and data storage are **Distributed**!

- **Business pattern i**
  - How to earn money!
- **Then, to carry out the “long tail”, we need the support of large scale data (products, customers, …) and computing power**
  - You have to consider to do business with the whole world! – of course in many stages
- **So many –**
  - Customers, Computers, Data,
  - with real-time processing request



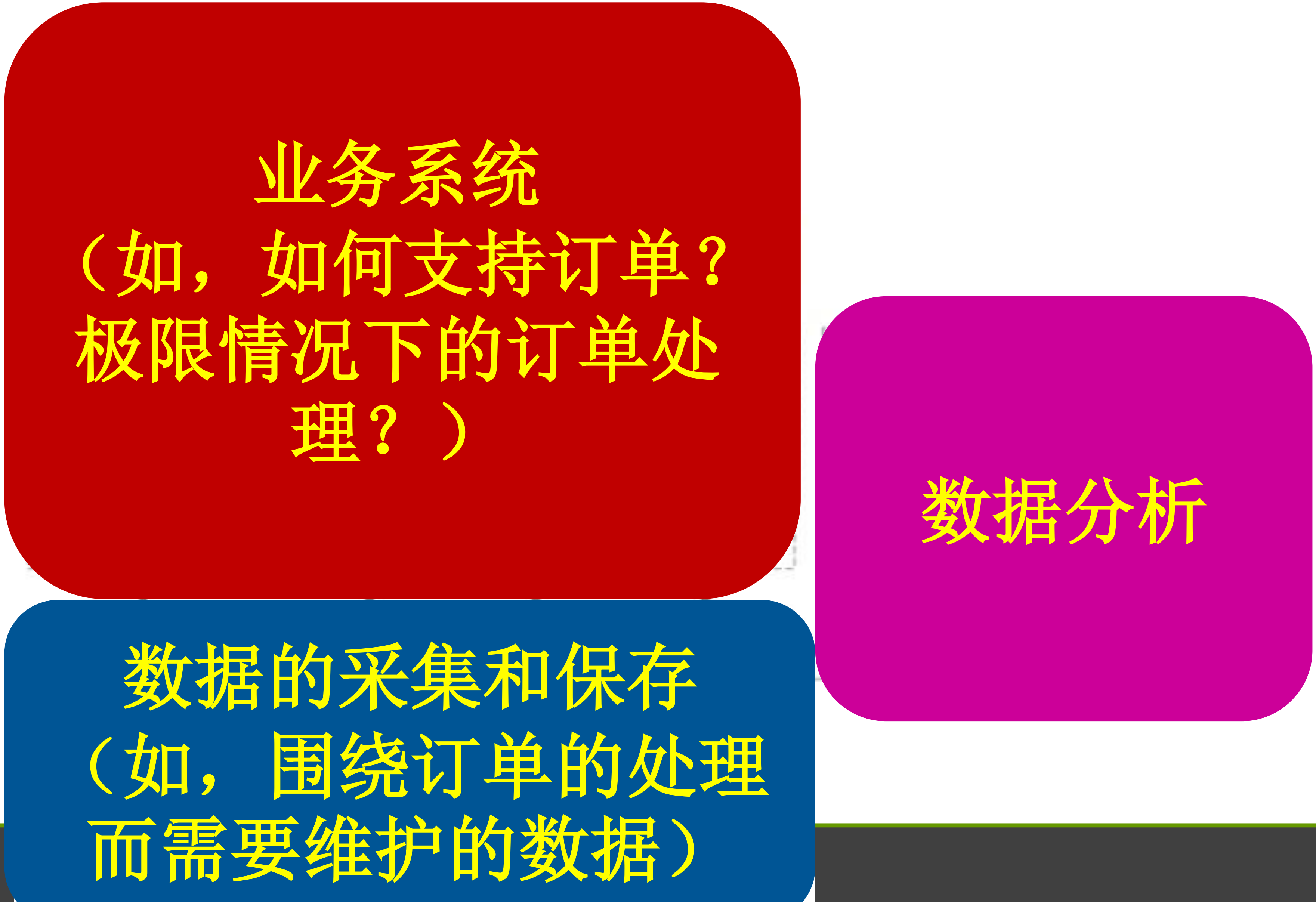
业务系统(Business System) 是立身之本~

业务系统
（如，如何支持订单？极限情况下的订单处理？）

数据分析

数据的采集和保存
（如，围绕订单的处理而需要维护的数据）

业务系统(Business System) 是立身之本~

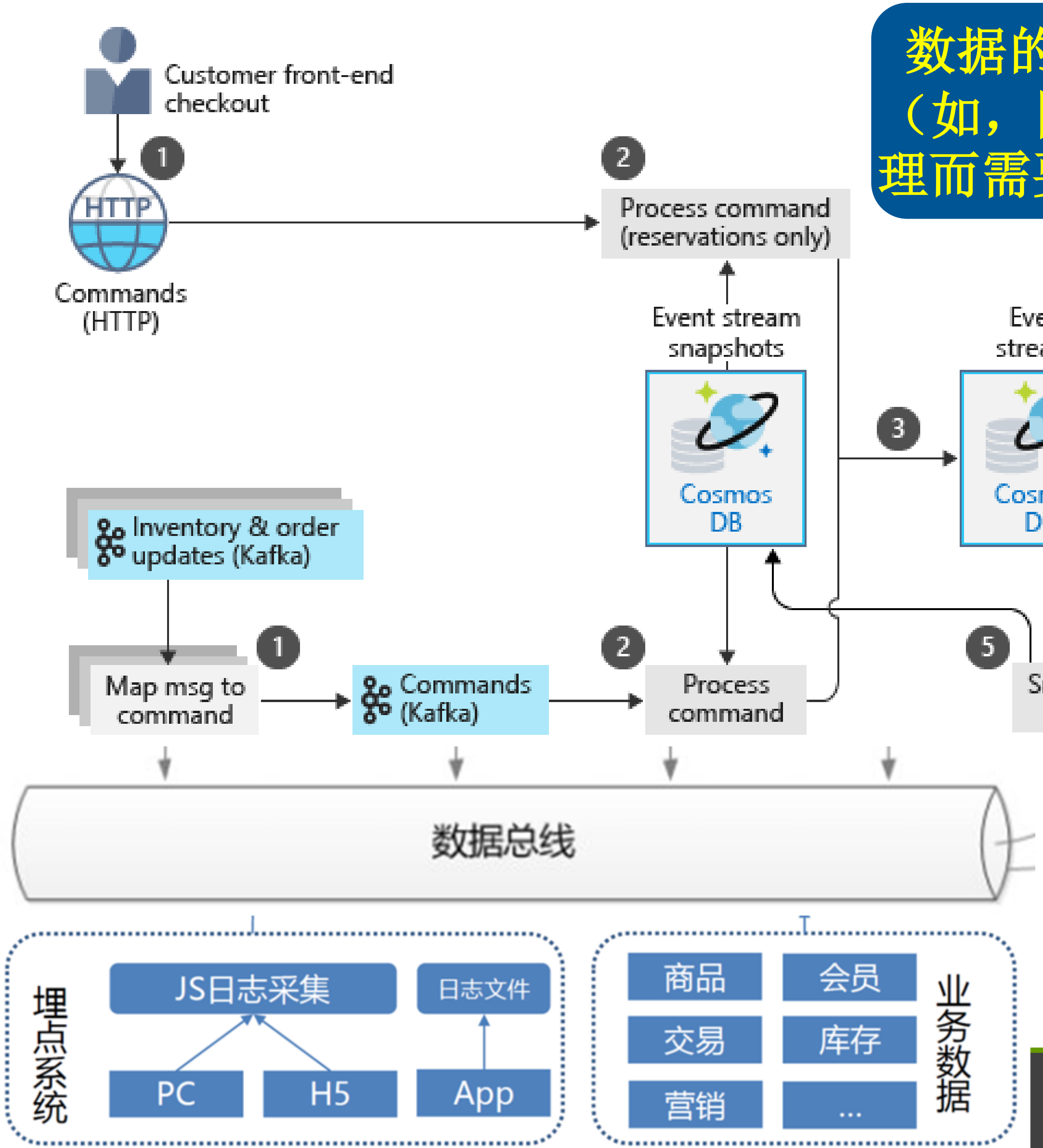


业务系统
（如，如何支持订单？极限情况下的订单处理？）

数据分析

数据的采集和保存
（如，围绕订单的处理而需要维护的数据）

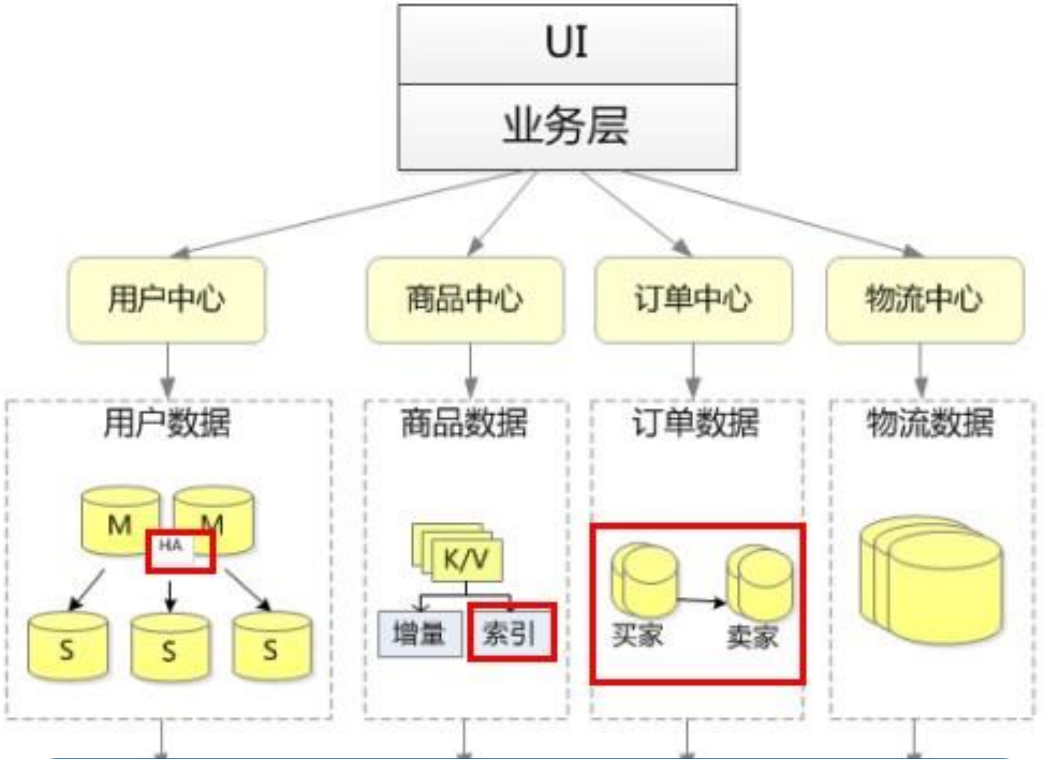
Data Collecting in Big Data era

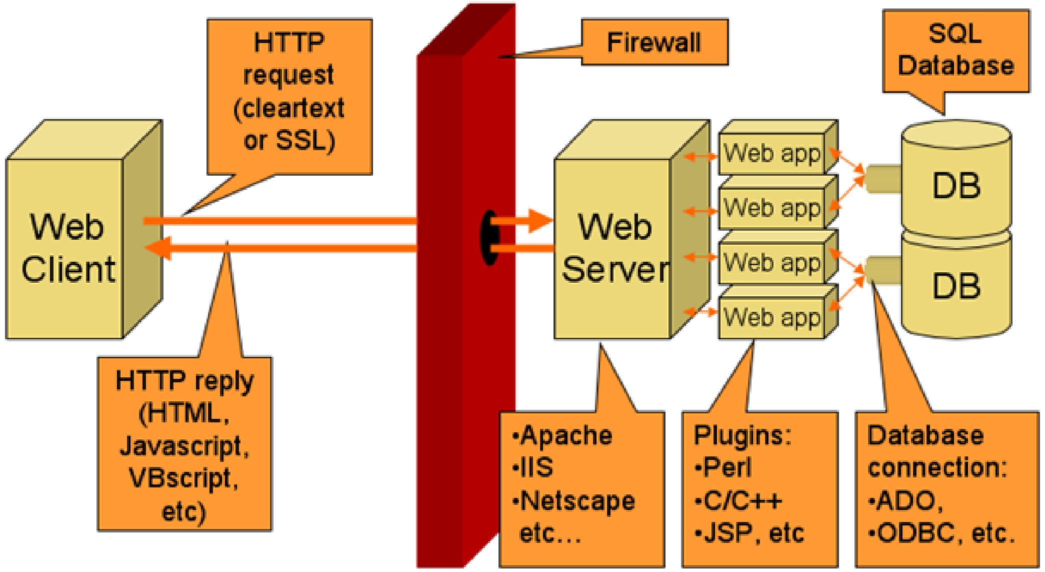


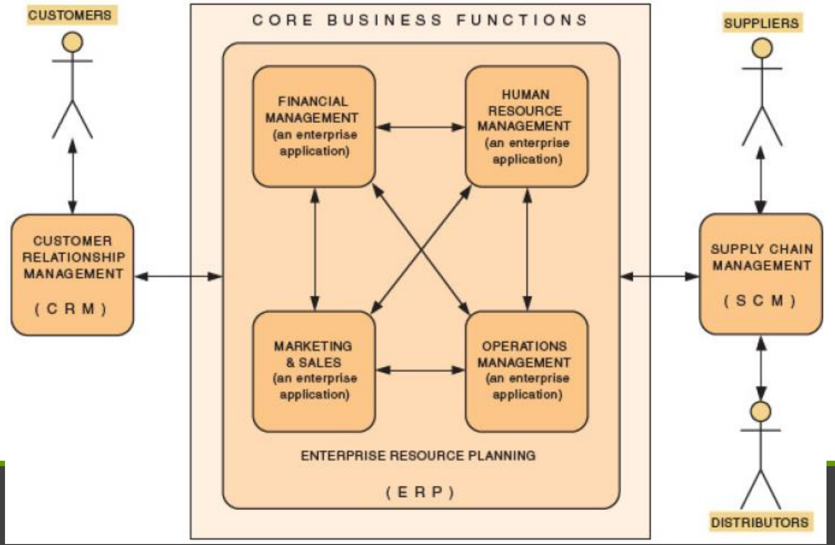
数据的采集和保存（如，围绕订单的处理而需要维护的数据）

- In Big Data era, many specific programs are proposed, and used to work together for business
- OMS – Order Management System
  - Streaming processing
  - Every actions of customers are kept in log [日志]
- BI – Business Intelligence
  - Recommendation, Advertisement, DM, …

Business System is definitely the basis/kernel!

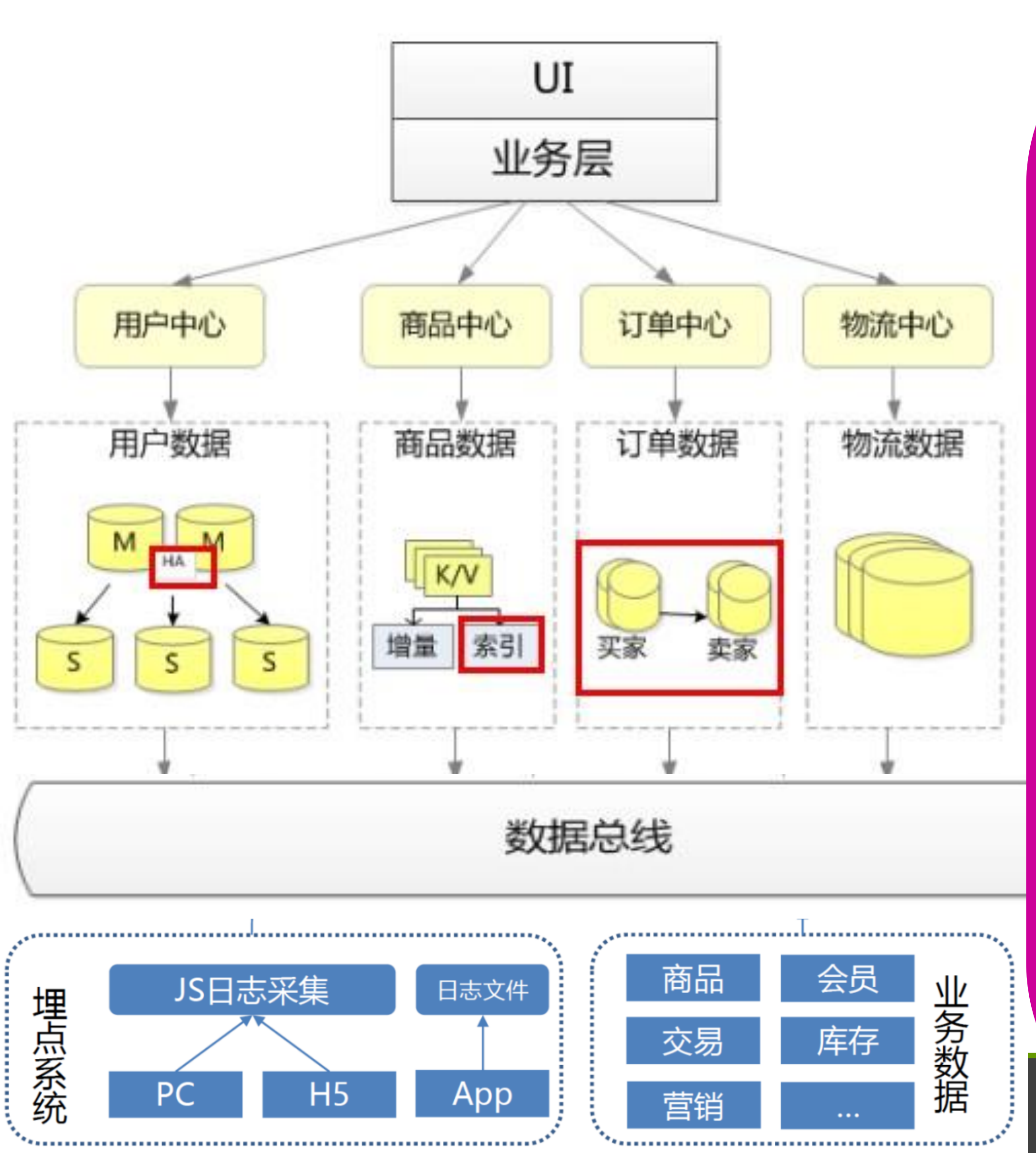





CUSTOMERS
CORE BUSINESS FUNCTIONS
SUPPLIERS
FINANCIAL MANAGEMENT (an enterprise application)
HUMAN RESOURCE MANAGEMENT (an enterprise application)
CUSTOMER RELATIONSHIP MANAGEMENT (CRM)
SUPPLY CHAIN MANAGEMENT (SCM)
MARKETING & SALES (an enterprise application)
OPERATIONS MANAGEMENT (an enterprise application)
ENTERPRISE RESOURCE PLANNING (ERP)
DISTRIBUTORS

数据的采集和保存
（如，围绕订单的处理
而需要维护的数据）

Big Analytics with DW



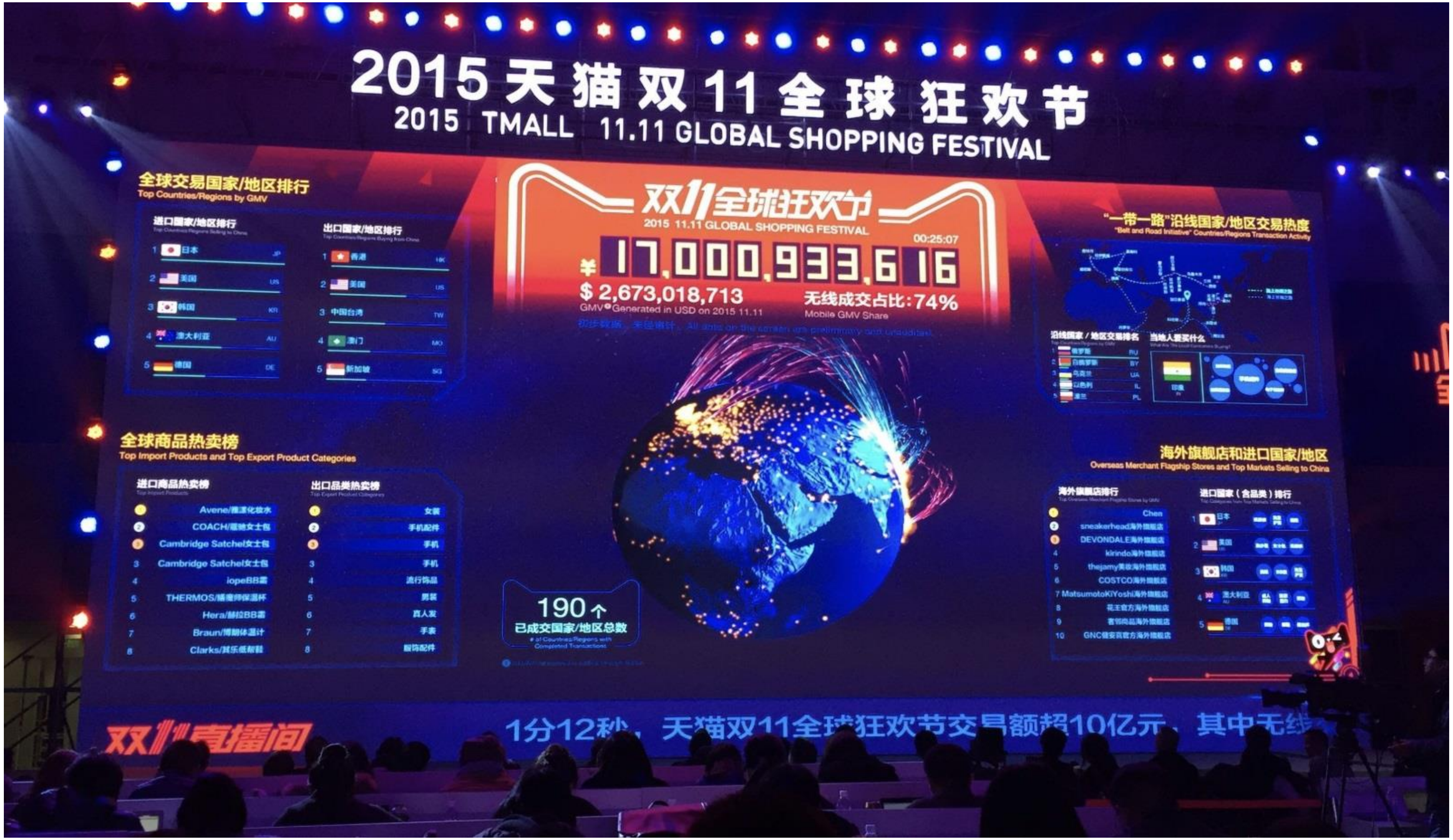
数据分析

Big Analytics in Big Data era

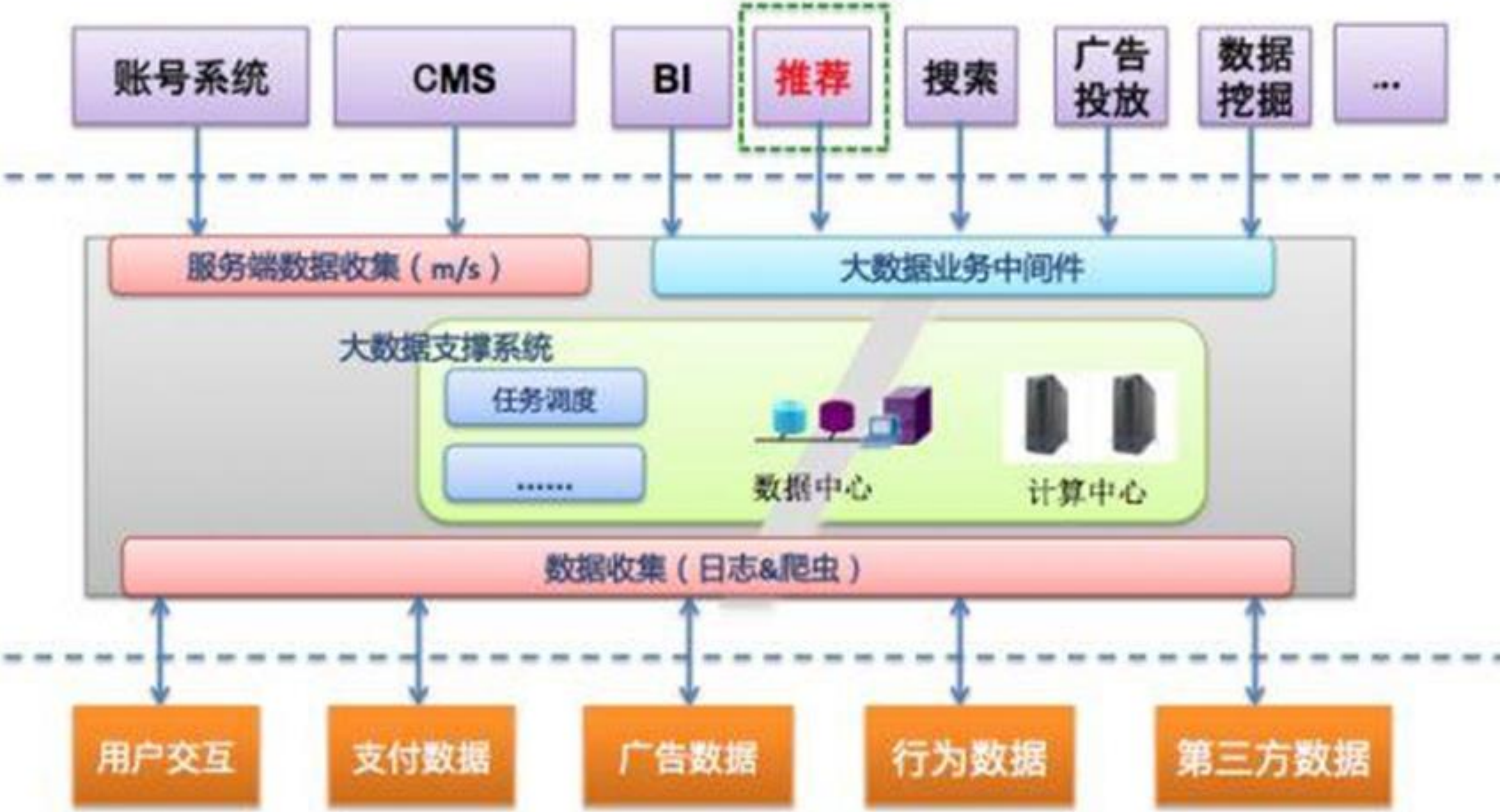
数据分析

数据的采集和保存
（如，围绕订单的处理而需要维护的数据）

Two valuable challenges – HUGE concurrency


2015 天猫双11全球狂欢节
2015 TMALL 11.11 GLOBAL SHOPPING FESTIVAL
全球交易国家/地区排行
Top Countries/Regions by GMV
进口国家/地区排行
出口国家/地区排行
双11全球狂欢节
2015 11.11 GLOBAL SHOPPING FESTIVAL
00:25:07
¥17,000,933,616
$ 2,673,018,713
GMV Generated in USD on 2015 11.11
无线成交占比:74%
Mobile GMV Share
“一带一路”沿线国家/地区交易热度
"Belt and Road Initiative" Countries/Regions Transaction Activity
全球商品热卖榜
Top Import Products and Top Export Product Categories
进口商品热卖榜
出口品类热卖榜
190个
已成交国家/地区总数
海外旗舰店和进口国家/地区
Overseas Merchant Flagship Stores and Top Markets Selling to China
双11直播间
1分12秒，天猫双11全球狂欢节交易额超10亿元，其中无线

Two valuable challenges – Targeting/Precision Advertising



https://www.infoq.cn/article/V0wtItKL7BAIyJ-WKoz8

大商务，需要大数据

- **In IT age, <u>platform sticking consumers</u> [黏着客户的平台] is the popular pattern for e-business – Amazon, Google, Alibaba, JD, …**
  - **Large scale** (data and computing power) is important, which needs HPC
- **Large Scale Data – Big Data**
  - From File to Big Data
- **Large Scale computing power – High Performance Computing is now popular for business**
  - According to Top 500, MPP and Cluster
- **Additional bonus for Scientific Computing**
  - Weather forecasting

Data Management in short ← My understanding

- **Data Management belongs to a larger framework of the interaction between human and the world**
  - Information Representation (Data structures):
    - How do we capture information and represent it?
  - Storage media
    - What media is used to store?
  - Processing (I.D.U.S)
    - How to carry out I.D.U.S operations?
  - Understanding (ML & Visualizing)
    - How to understand information and find rules [规律]?

Before IT

hieroglyphist [ˌhaiərəˈglifist]
n. 象形文字研究者,书写象形文字者

Info. Represent (Data)	Store	Processing (I.D.U.S)	Understanding & Visualizing
	Mud board	US - Human with hands and tools	US - Human with brains
cave painting	Stone		
	Oracle-bone		

Drawings and ancient characters

Before IT

Info. Represent (Data)	Store	Processing (I.D.U.S)	Understanding & Visualizing
	Bamboo slips	US - Human with hands and tools	US - Human with brains
	Silk		
	Paper files		

In IT

Info. Represent (Data)	Store	Processing (I.D.U.S)	Understanding & Visualizing
Structured Data	A file system	Specific programs	Statistics
	RDBMS/ Business Sys/files	SQL	Statistics
Data is diverse	Data Warehouse / RDBMS/files	SQL+MDDM operations	Statistics, DM, ML,…
Data is Huge & Diverse	Big Data/files	Map/Reduce SQL Streaming …	Statistics, DM, ML,…

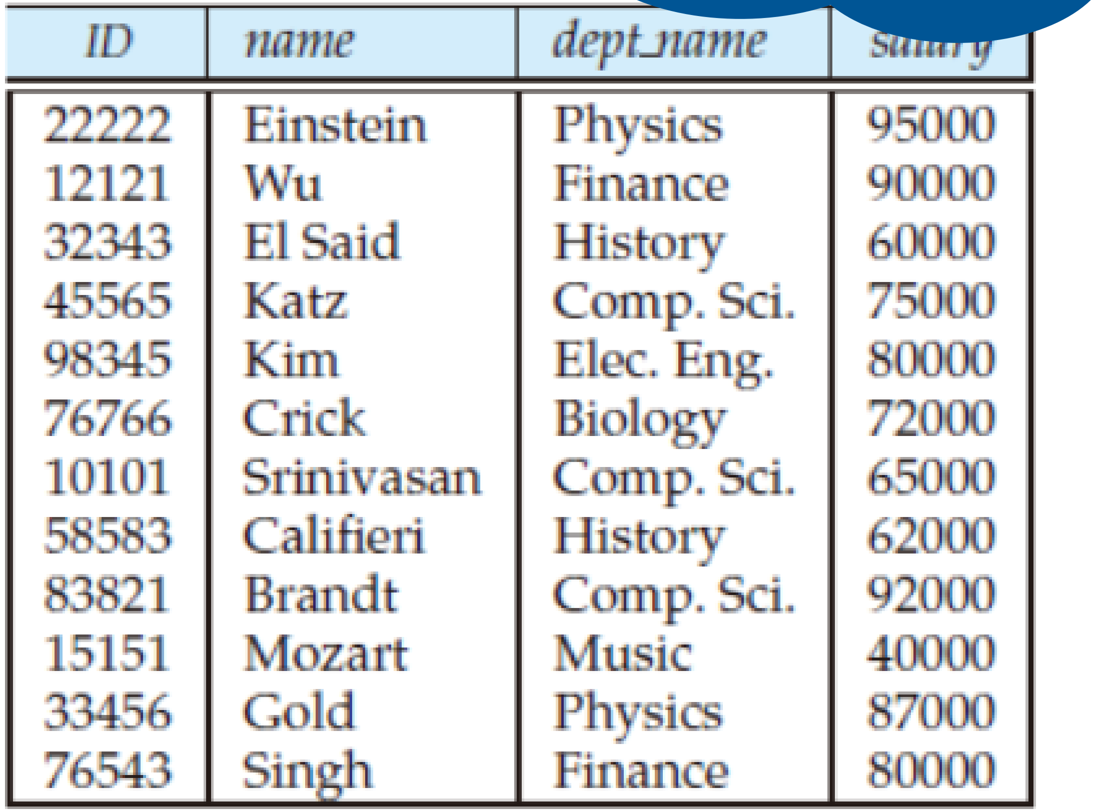
DATA MANAGEMENT now in IT age
FILE is the basis

- **FILE is the basis**
  - Device drivers encapsulate the “*evil*” details of physical devices and provide uniform APIs for programmers to manage data easily – stored as files in permanent storage media
  - APIs – C Interface
    - # include <stdio.h>
    - FILE *fopen(char *path, char *type);
      - ✓ int feof(FILE *stream);
      - ✓ int fseek(FILE *stream, long offset, int origin);
      - ✓ int fscanf(FILE *stream, const char *format, ...);
      - ✓ int fprintf(FILE *stream, const char *format, ...);
    - int remove(char *path);
    - int fclose(FILE *fp);

https://en.wikipedia.org/wiki/C_file_input/output

Here with C – Data M

You can finish I.D.U.S based on those IO functions – old way



ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music	40000
33456	Gold	Physics	87000
76543	Singh	Finance	80000

(a) The *instructor* table

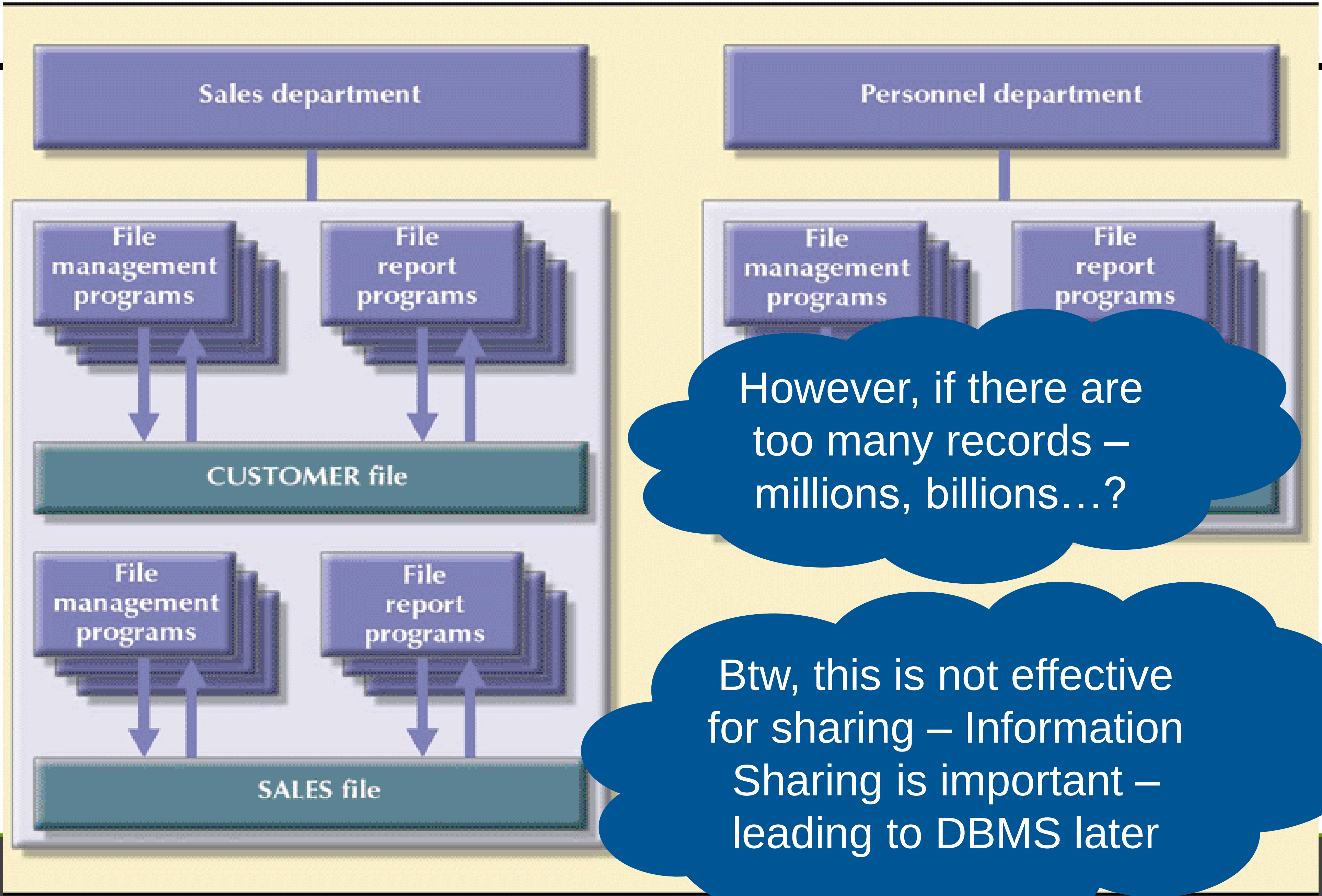
you know

- How to store the data shown in left?
- You define
  - define struct{

```
    int id;
    string name;
    string dept_name;
    int salary;
} instructor;
```

- Use some IO functions
  - scanf()
  - printf()
  - seek()…

A Simple File System



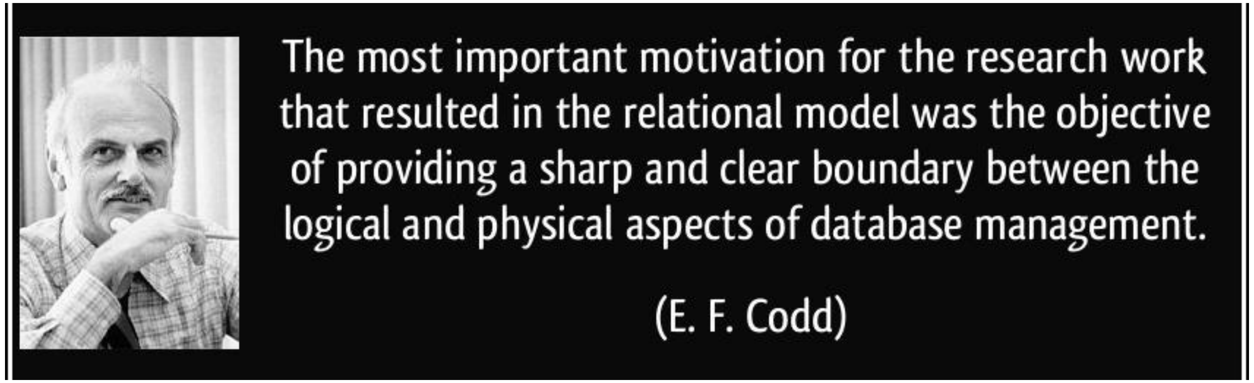
However, if there are too many records – millions, billions…?

Btw, this is not effective for sharing – Information Sharing is important – leading to DBMS later

Relational DBMS

Born: August 23, 1923
Died: April 18, 2003 (aged 79)

□ 1970 - E.F. Codd and the Relational Model



The most important motivation for the research work that resulted in the relational model was the objective of providing a sharp and clear boundary between the logical and physical aspects of database management.

(E. F. Codd)

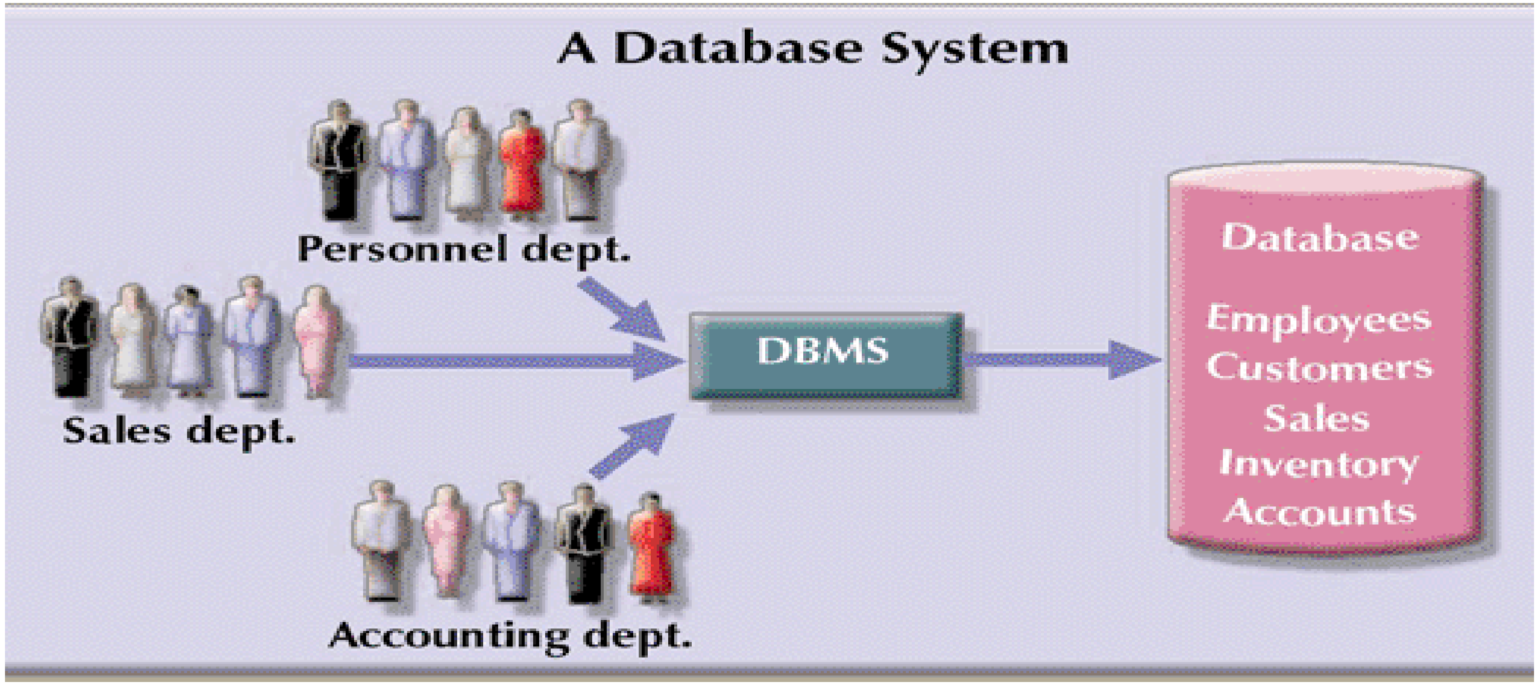
- I prefer to use to the 1 top goal and 2 roles to indicate the benefit of DBMS
  - ➢1 Top goal:
    - ✓Support the **data access** of **many users**
  - ➢2 roles:
    1. Concurrent data management by many use
    2. Provide friendly/flexible interaction for common users

SQL processing is one kernel of RDBMS.
Do you know How?

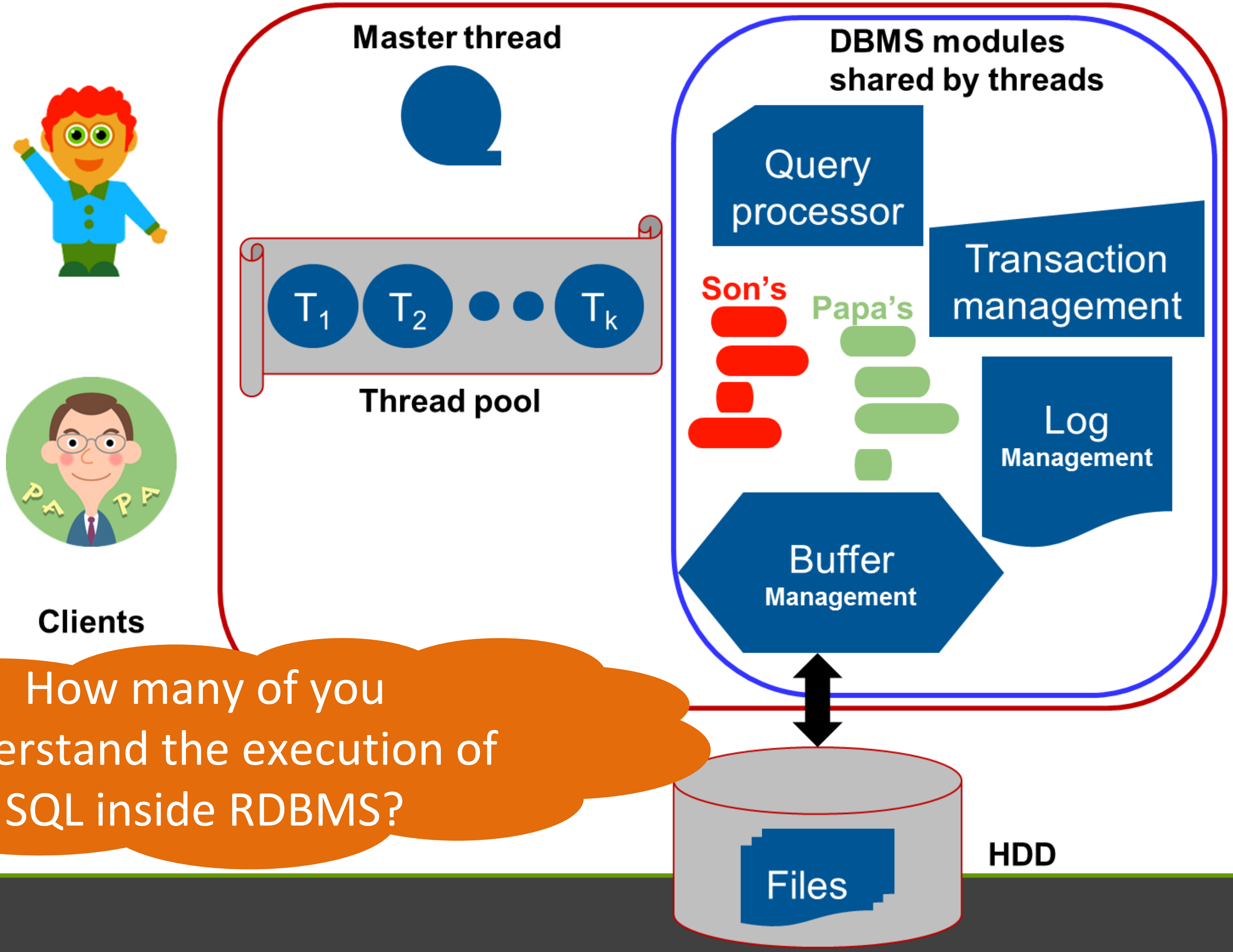
To show **How to ensure providing services for diverse braches** is the core of DBMS and this PP

Database vs. File System

- Problems inherent in file systems make usi em desirable
- File system: Many separate and unrelated files
- Database : Logically related data stored in a single logical data repository

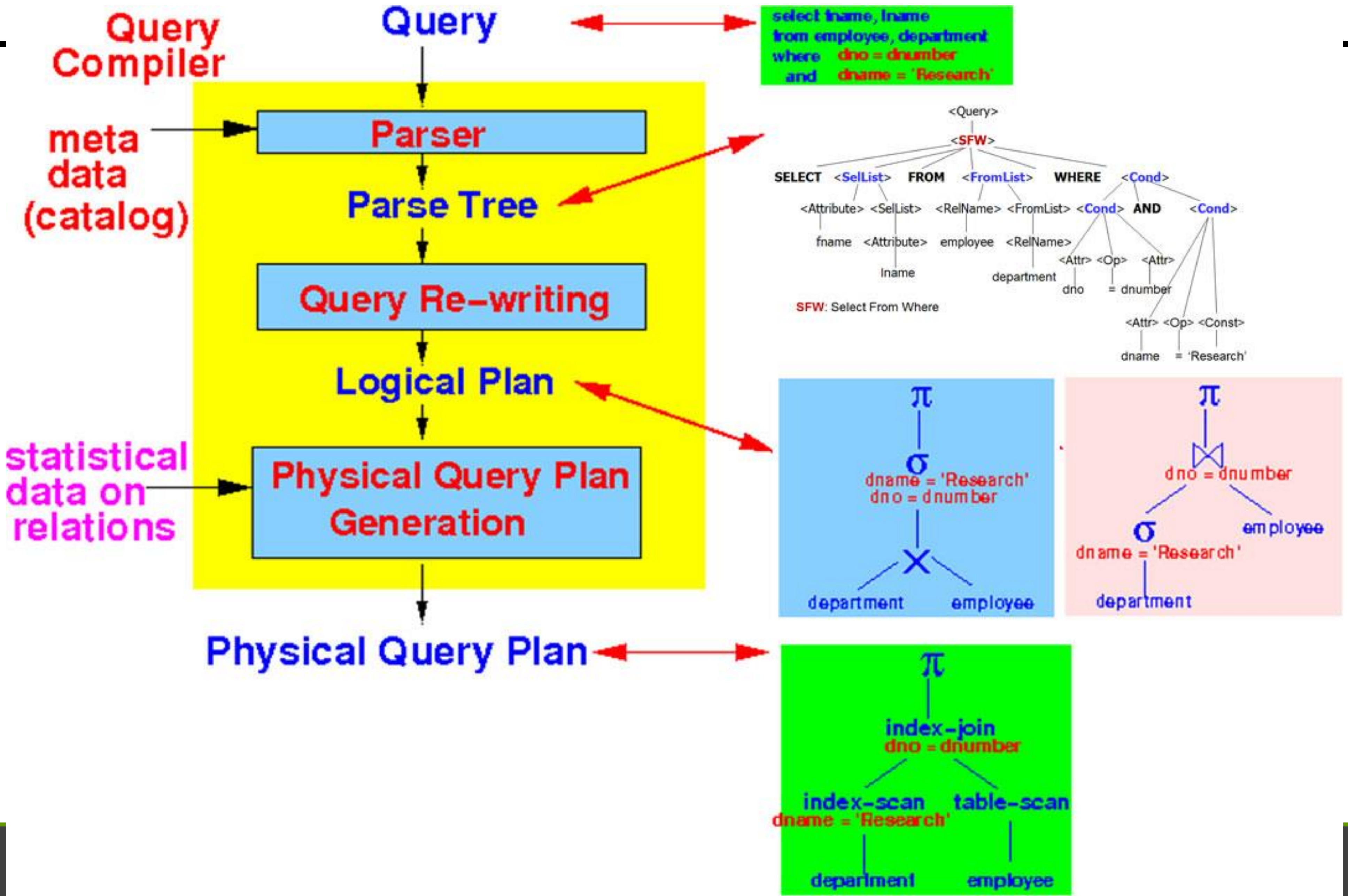


Sketch of internal modules of modern DBMS



How many of you understand the execution of SQL inside RDBMS?

In short



Query Compiler

Query

```
select fname, lname
from employee, department
where   dno = dnumber
  and   dname = 'Research'
```

meta data (catalog)

Parser

Parse Tree

<Query>
<SFW>
SELECT <SelList> FROM <FromList> WHERE <Cond>
<Attribute> <SelList> <RelName> <FromList> <Cond> AND <Cond>
fname <Attribute> employee <RelName>
<Attr> <Op> <Attr>
lname
department
dno = dnumber
<Attr> <Op> <Const>
dname = 'Research'

SFW: Select From Where

Query Re-writing

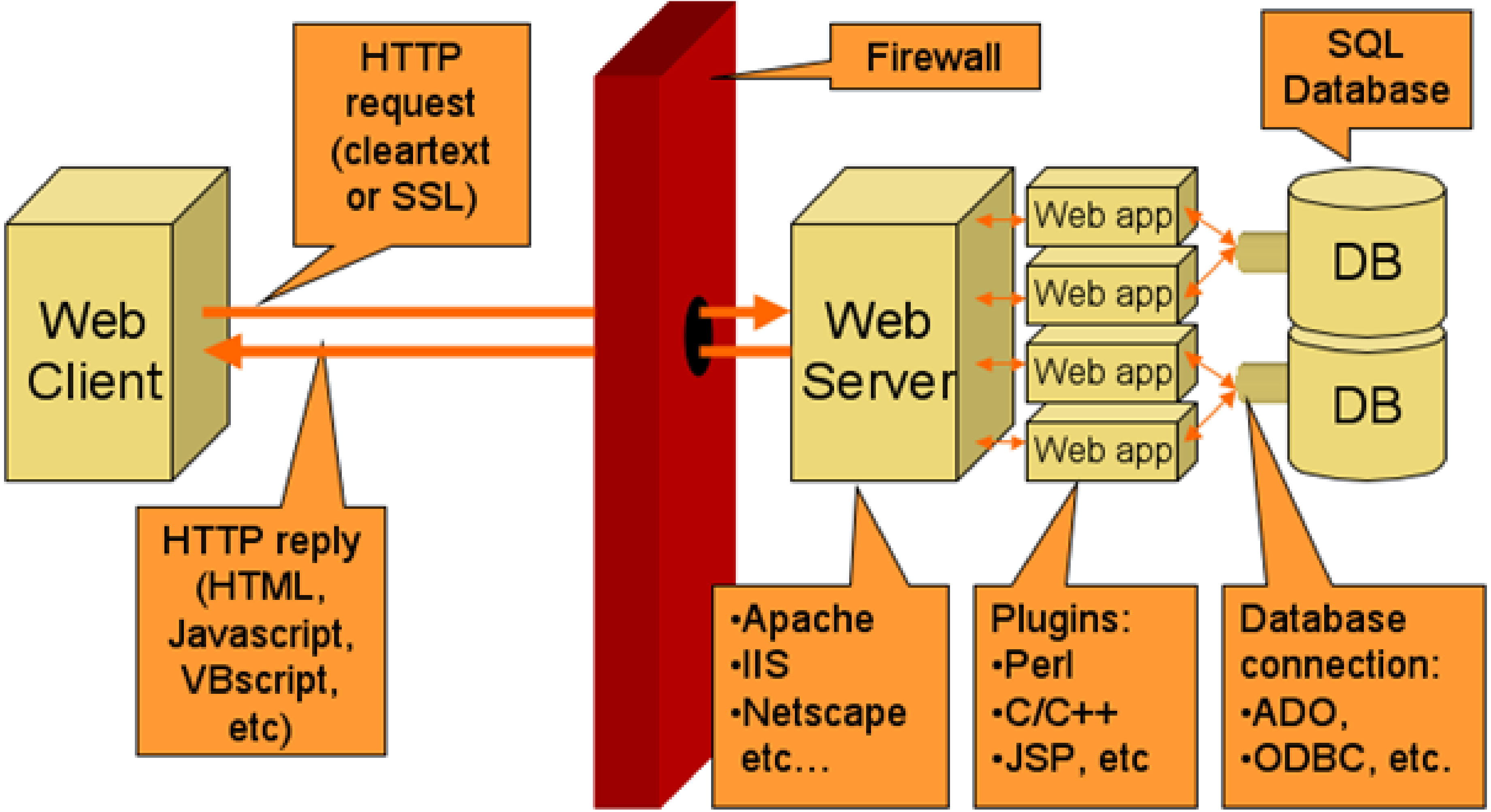
Logical Plan

statistical data on relations

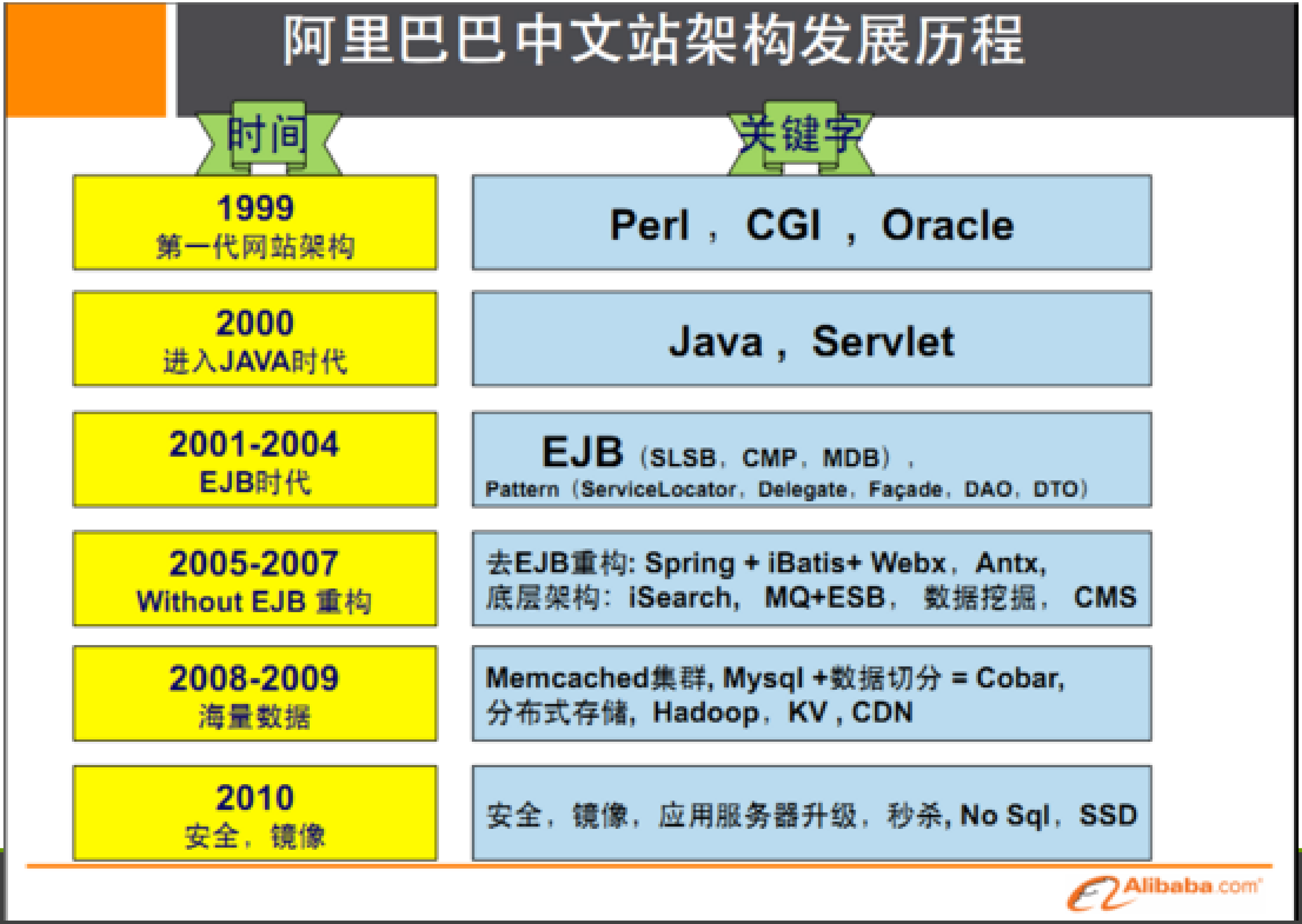
Physical Query Plan Generation

Physical Query Plan

- Now many techniques for dynamic – JSP is one of them



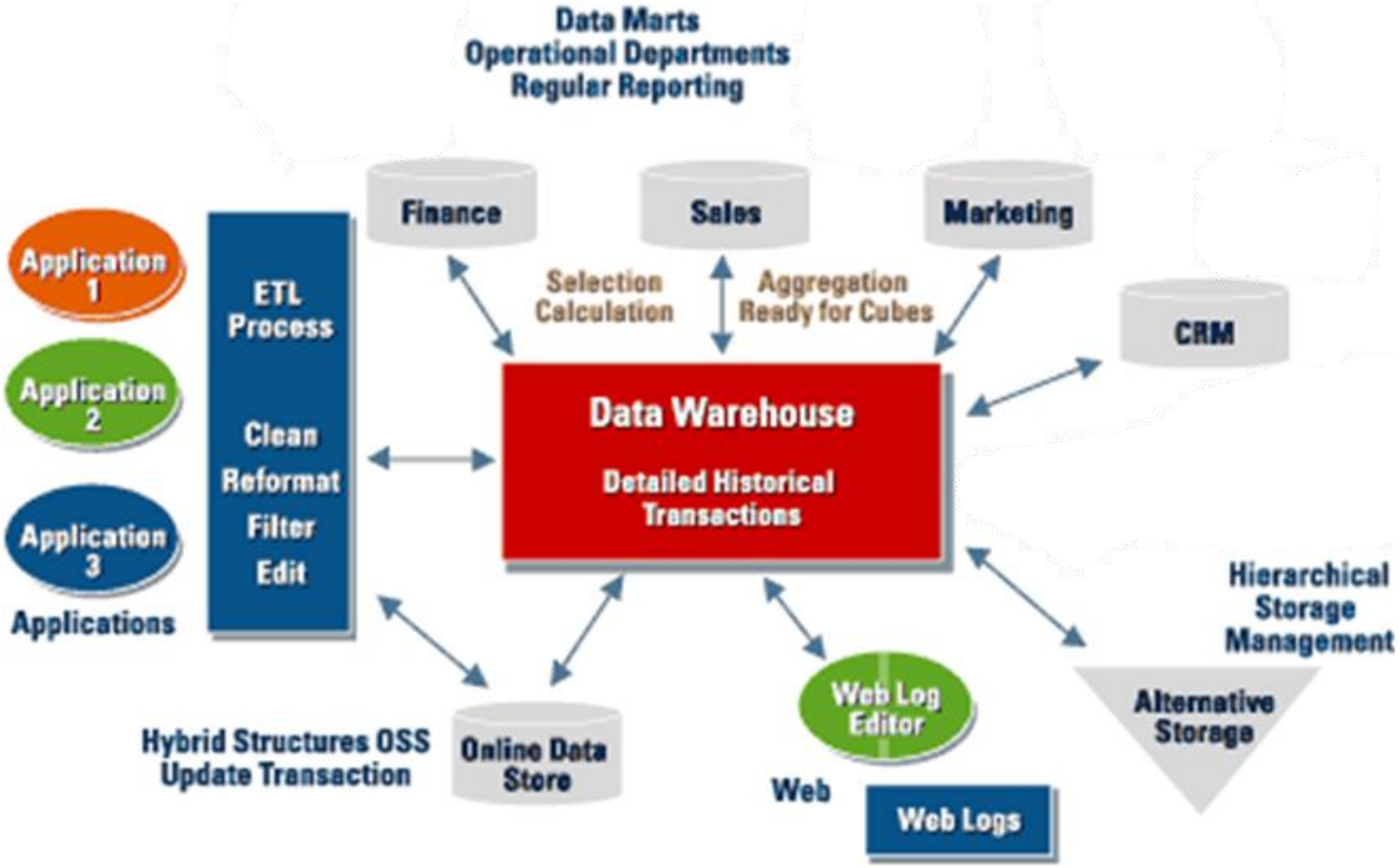
https://www.codercto.com/a/30702.html



阿里巴巴中文站架构发展历程

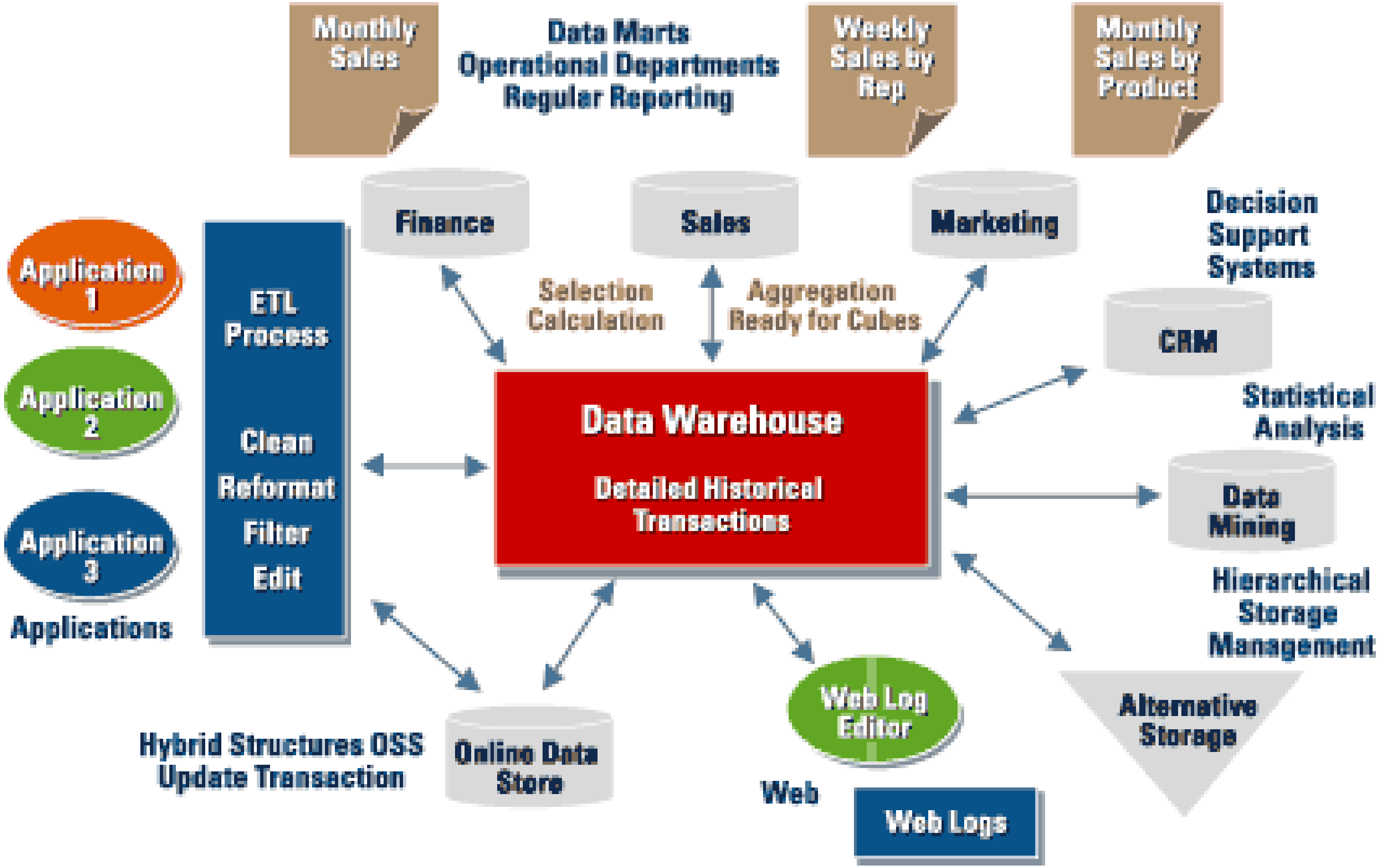
时间	关键字
1999 第一代网站架构	Perl，CGI ，Oracle
2000 进入JAVA时代	Java，Servlet
2001-2004 EJB时代	EJB（SLSB、CMP、MDB）， Pattern（ServiceLocator、Delegate、Façade、DAO、DTO）
2005-2007 Without EJB 重构	去EJB重构: Spring + iBatis+ Webx，Antx, 底层架构：iSearch， MQ+ESB， 数据挖掘， CMS
2008-2009 海量数据	Memcached集群, Mysql +数据切分 = Cobar, 分布式存储, Hadoop，KV , CDN
2010 安全，镜像	安全，镜像，应用服务器升级，秒杀, No Sql，SSD

And Data Warehouse comes – Manage data first



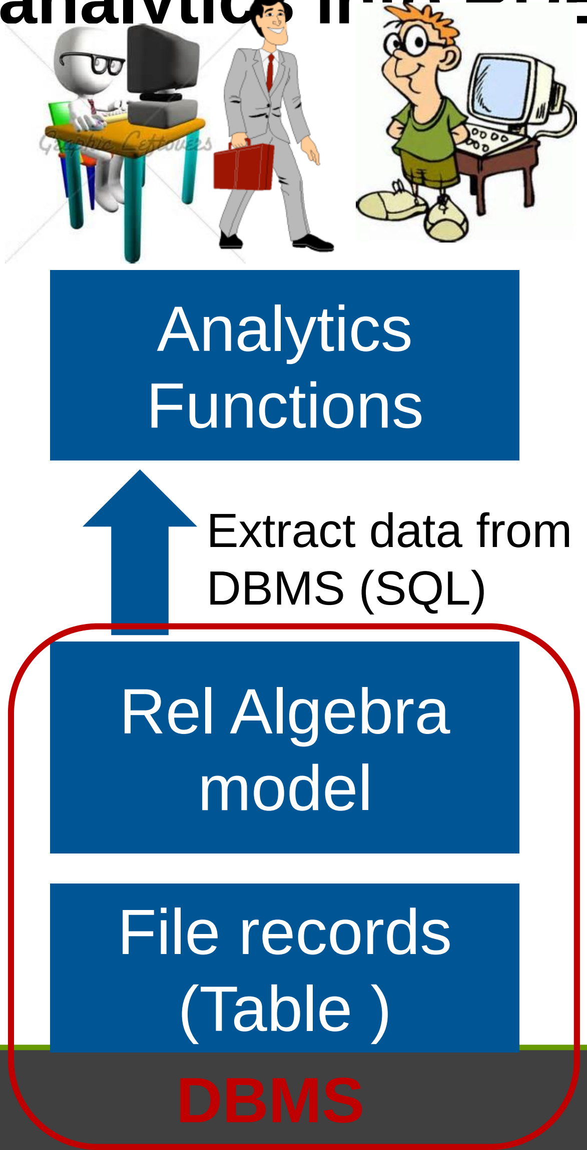
http://download.101com.com/pub/tdwi/images/ww20/Syncsort-dia1.gif

DW also tries to integrate Data Analytics



By Data Warehouse
– Integrating Data management and processing

- □ **Integrate Data analytics into RDBMS was tried in 1990s**



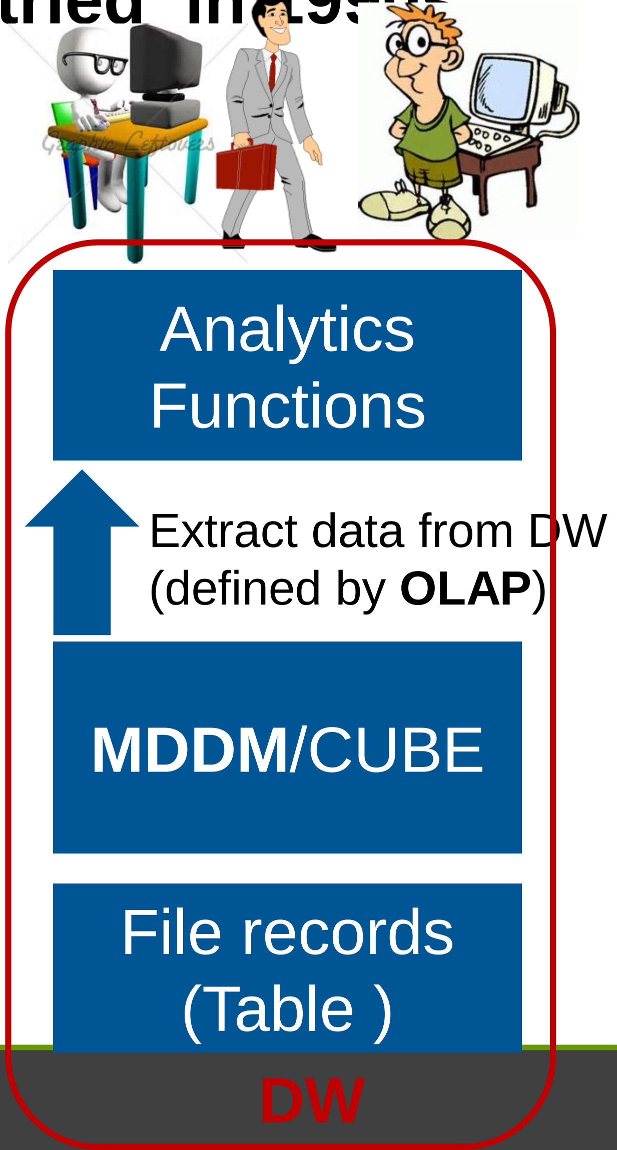
Analytics Functions

Extract data from DBMS (SQL)

Rel Algebra model

File records (Table)

DBMS



Analytics Functions

Extract data from DW (defined by **OLAP**)

MDDM/CUBE

File records (Table)

DW

We need ANOVA (ANalysis Of VAriance) here

- **The logic for ANOVA**
  - The ANOVA test is based on the **combined distances** from $\bar{\bar{x}}$.
  - If the combined distances are **large**, that indicates we should reject H_0.
  - Statistical variable for ANOVA
    - **SSE** (Sum of Squared Errors)
      - ✓ Or SSB(Sum of Squares Between groups)
    - **SSA** (Sum of Squared Average)
    - **MSE** (Mean Square Error)
    - **MSA** (Mean Square Average)

$$SSE = \sum_{i=1}^{k}\sum_{j=1}^{n_i}\left(x_{ij} - \bar{x}_i\right)^2$$

$$SSA = \sum_{i=1}^{k}\sum_{j=1}^{n_i}\left(\bar{x}_i - \bar{\bar{x}}\right)^2 = \sum_{i=1}^{k} n_i\left(\bar{x}_i - \bar{\bar{x}}\right)^2$$

$$MSE = \frac{SSE}{n-k}$$

$$MSA = \frac{SSA}{k-1}$$

“Trade space for time” [以空间换时间]

Table 2 Beverage sales of 4 colors and their means

Market (*j*)	Group *A* (*i*)				
	No color (A_1)	Pink(A_2)	saffron yellow (A_3)	Green	
1	26.5	31.2	27.9	30.8	
2	28.7	28.3	25.1	29.6	
3	25.1	30.8	28.5	32.4	
4	29.1	27.9	24.2	31.7	
5	27.2	29.6	26.5	32.8	
合计	136.6	147.8	132.2	157.3	573.9
Group Mean # in group	$\bar{x}_1$ =27.32 n_1=5	$\bar{x}_2$=29.56 n_2=5	$\bar{x}_3$=26.44 n_3=5	$\bar{x}_4$=31.46 n_4=5	Grand mean $\bar{\bar{x}}$=28.695

We hope we could do business analysis quickly on HUGE data, and this means it’s better to store cumulative information in advance

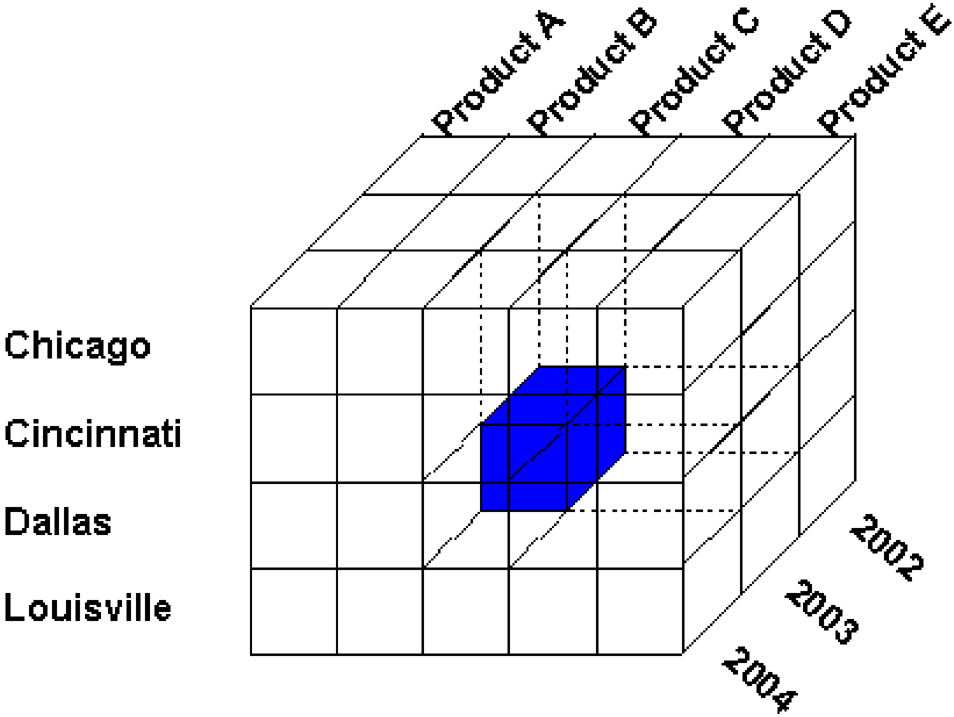
New model is needed – MDDM/CUBE

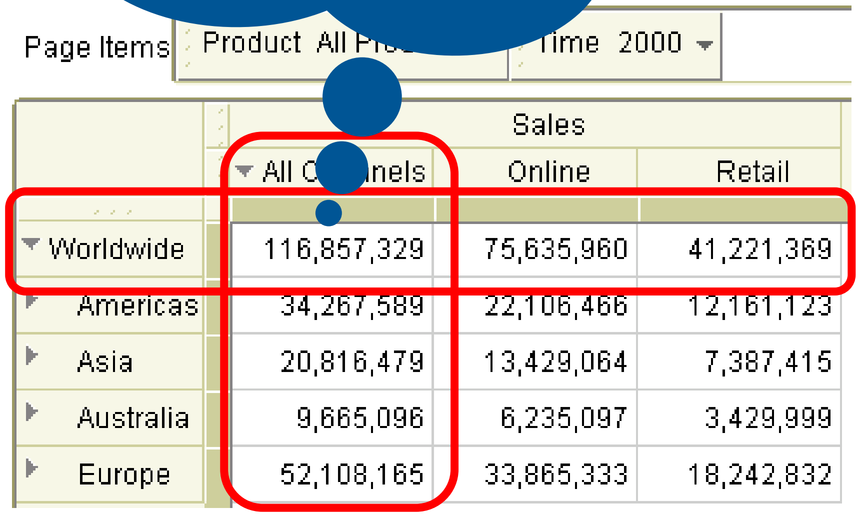
http://docs.oracle.com/html/B13915_04/i_olap_chapter.htm

Define the data with the multi-dimensional view

- A multidimensional data is optimized for a ata. Such data sources are sometimes calle analytical processing (OLAP) data so

Some cumulative information is stored in advanced and will be updated always





Page Items | Product All Pro | Time 2000

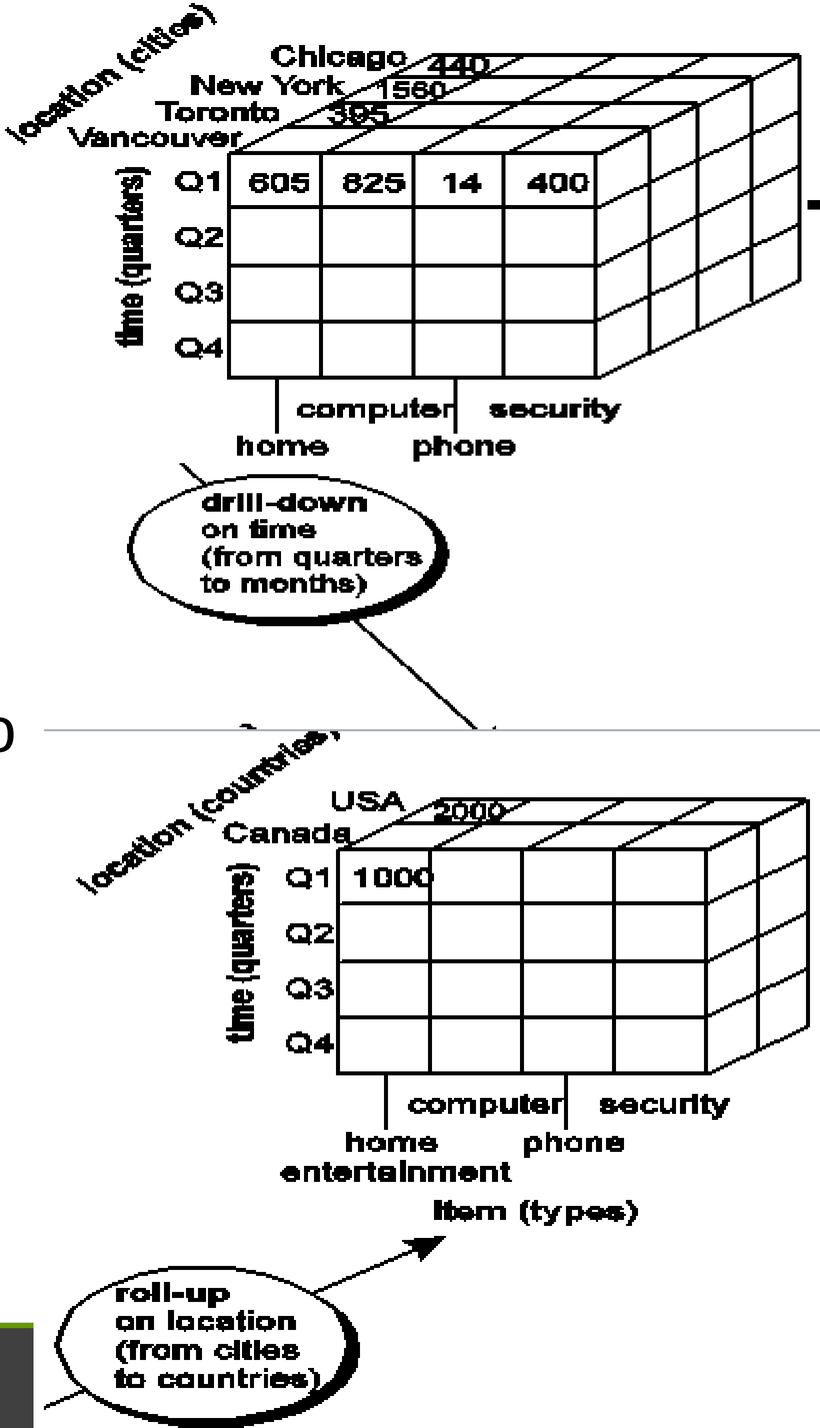
	Sales		
	All C nels	Online	Retail
Worldwide	116,857,329	75,635,960	41,221,369
Americas	34,267,589	22,106,466	12,161,123
Asia	20,816,479	13,429,064	7,387,415
Australia	9,665,096	6,235,097	3,429,999
Europe	52,108,165	33,865,333	18,242,832

Typical Operations on CUBE (For OLAP)

- Slice and dice: ***project and select***
- Roll up (drill-up): **summarize data**
  - *by climbing up hierarchy or by dimension reduction*
- Drill down (roll down): **reverse of roll-up**
  - *from higher level summary to lower level summary or detailed data, or introducing new dimensions*
- Pivot (rotate):
  - *reorient the cube, visualization, 3D to series of 2D planes*
- **Other operations**
  - *drill across: involving (across) more than one fact table*
  - *drill through: through the bottom level of the cube to its back-end relational tables (using SQL)*

Typical OLAP Operations

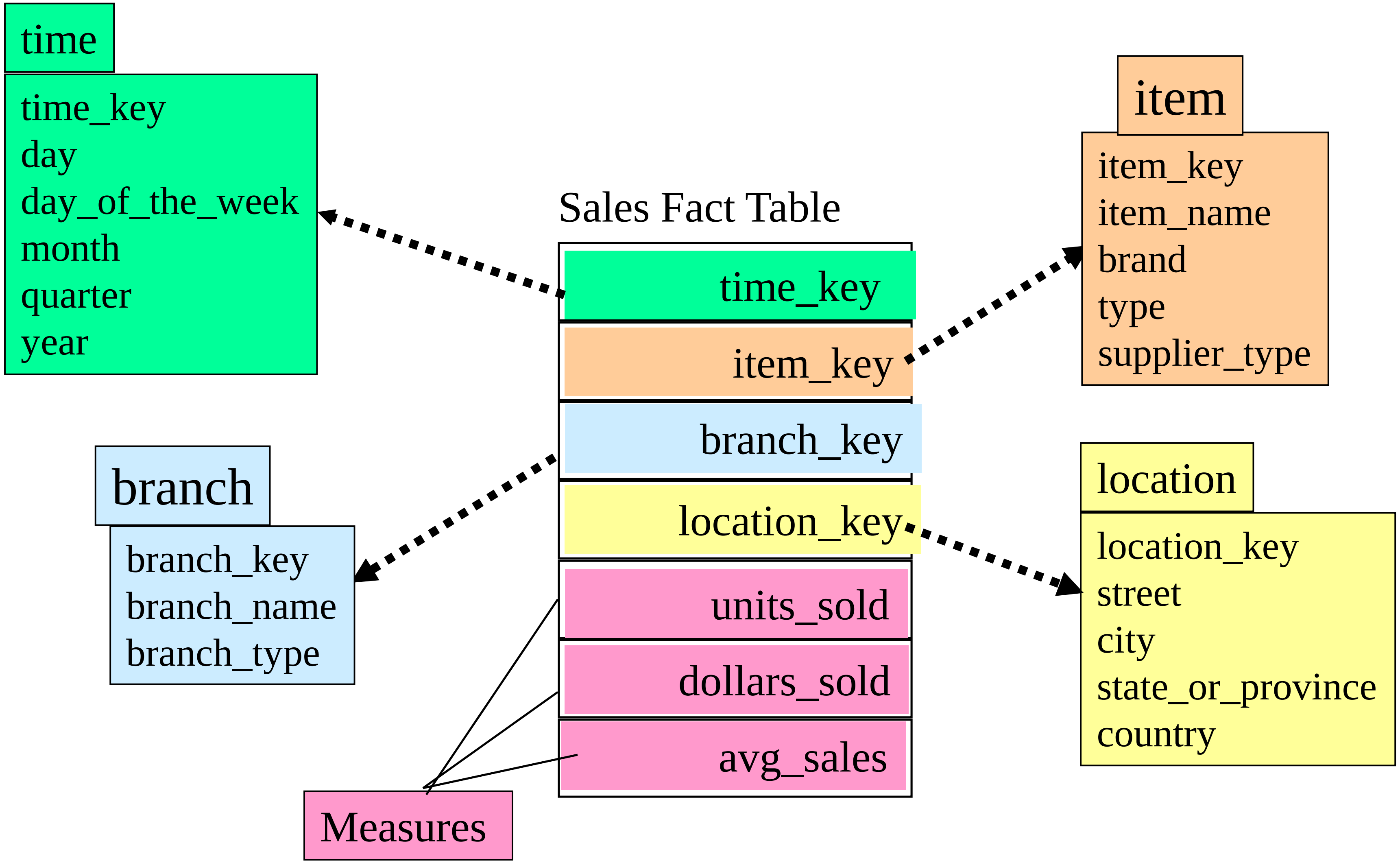
- **Cube is like the right**
  - It has 3 dimensions – time, location and item
- **By “drill down”, it means**
  - We have the total value, and want to know its organization
- **By “roll up”, it means**
  - To know its portion in higher level



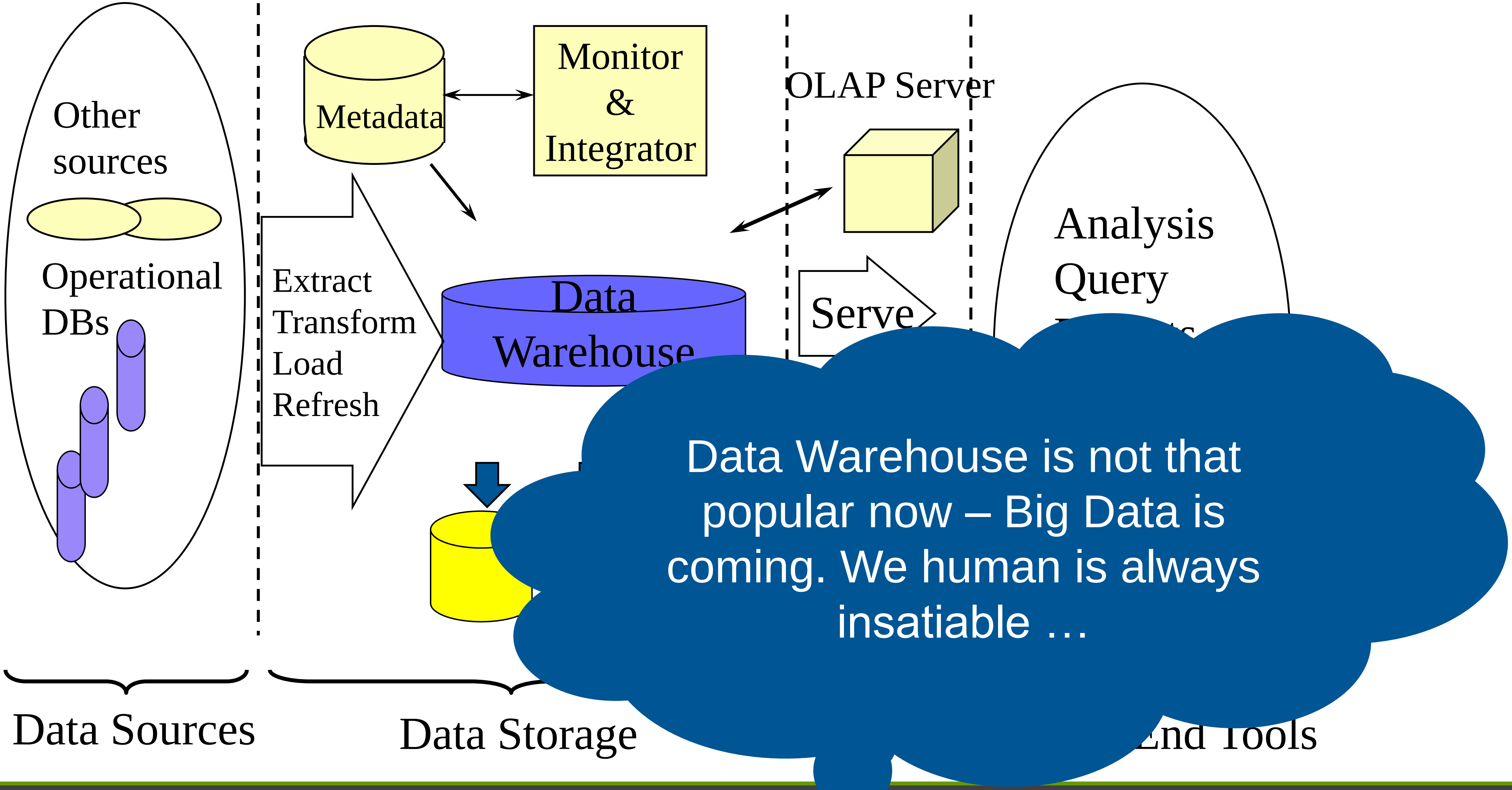
Implementation of Data Warehouses could be based on RDBMS

- **Modeling data warehouses: dimensions & measures**
  - Star schema: A fact table in the middle connected to a set of dimension tables
  - Snowflake schema: A refinement of star schema where some dimensional hierarchy is normalized into a set of smaller dimension tables, forming a shape similar to snowflake
  - Fact constellations [群体]: Multiple fact tables share dimension tables, viewed as a collection of stars, therefore called galaxy schema or fact constellation

MDDM of Star Schema (based on RDBMS)



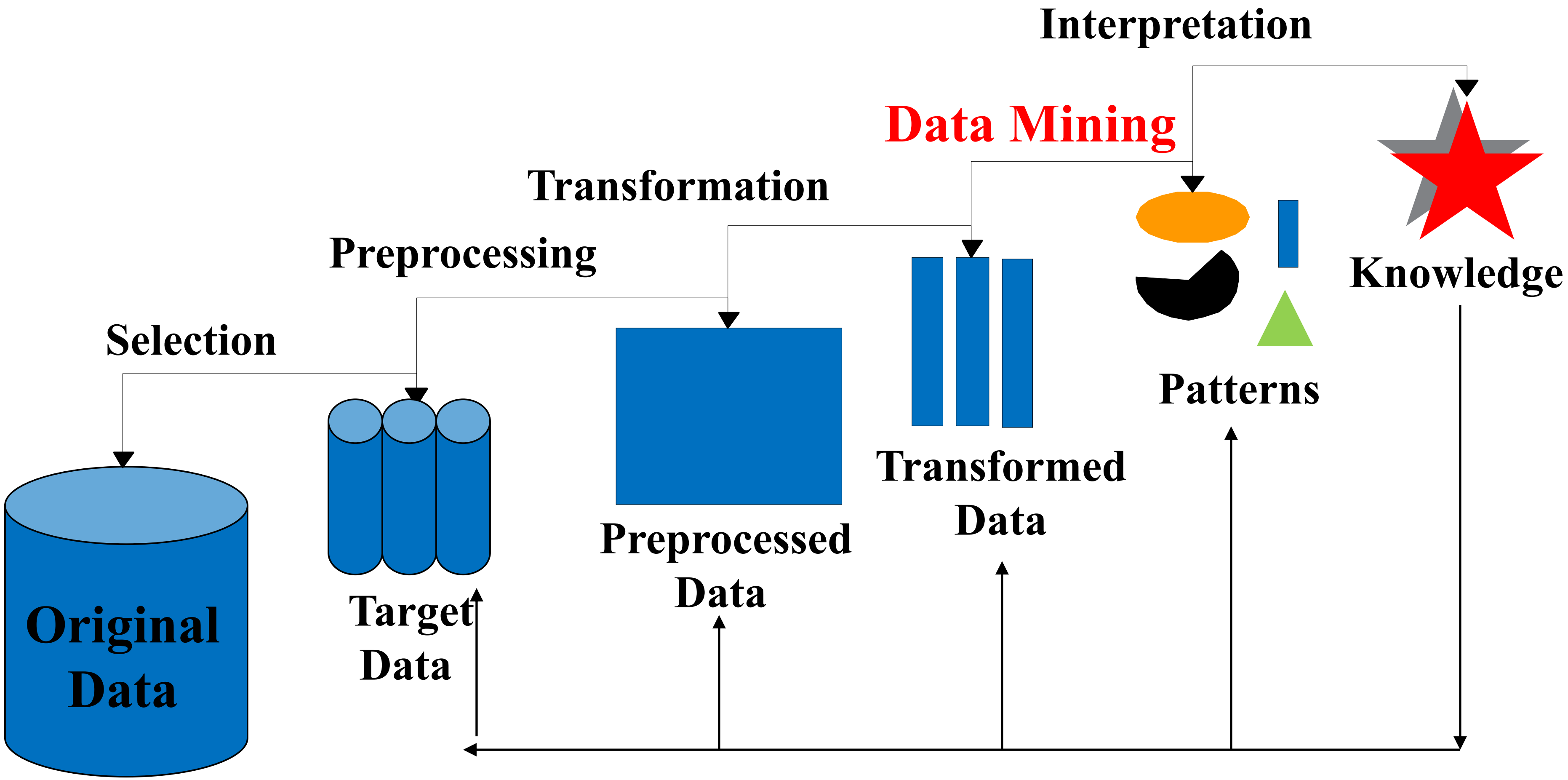
Data Warehouse
integrate analytics with data management



Data Warehouse is not that popular now – Big Data is coming. We human is always insatiable …

More popular model for Data Mining- KDD

- KDD: Knowledge Discovery from Data



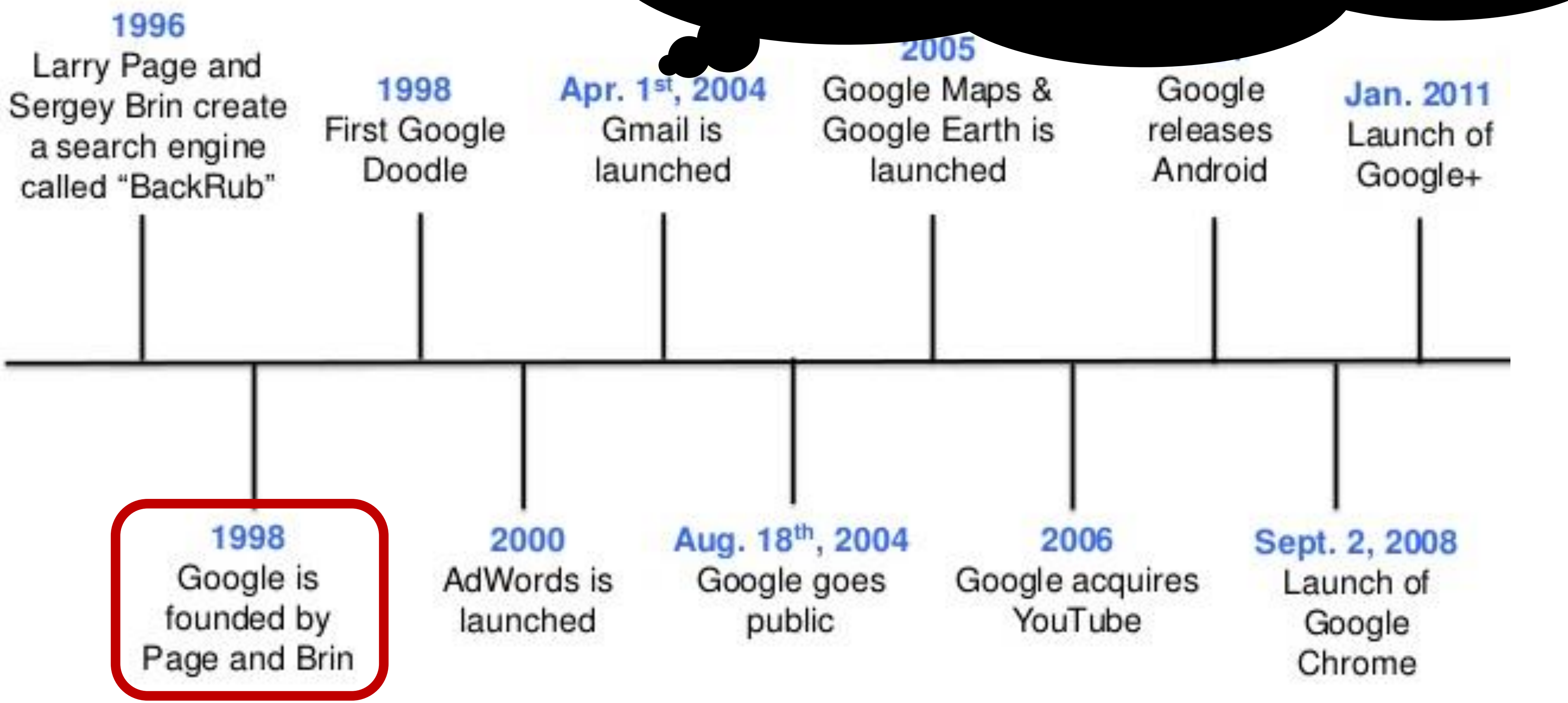
Big Data is coming

- **Ambitious to manage huge and diverse data – 3Vs: Volume, Variety, Velocity**
  - 2.5 quintillion bytes of data are generated every day!
    - ➢A quintillion is 10^{18}
  - Coming from many quarters, like Social media sites, Sensors, Digital photos, Business transactions, Location-based data, Web data, e-commerce, Bank/Credit Card, …
  - For information: Google processes 20 PB a day (2008)
    - ➢http://www.worldwidewebsize.com/
      - ✓“The Indexed Web contains **at least** **9.18 billion pages** (Sunday, 09 December, 2012).”
  - For science: NASA & Hubble scope

Google triggers other sea

- CBIR, XML, …

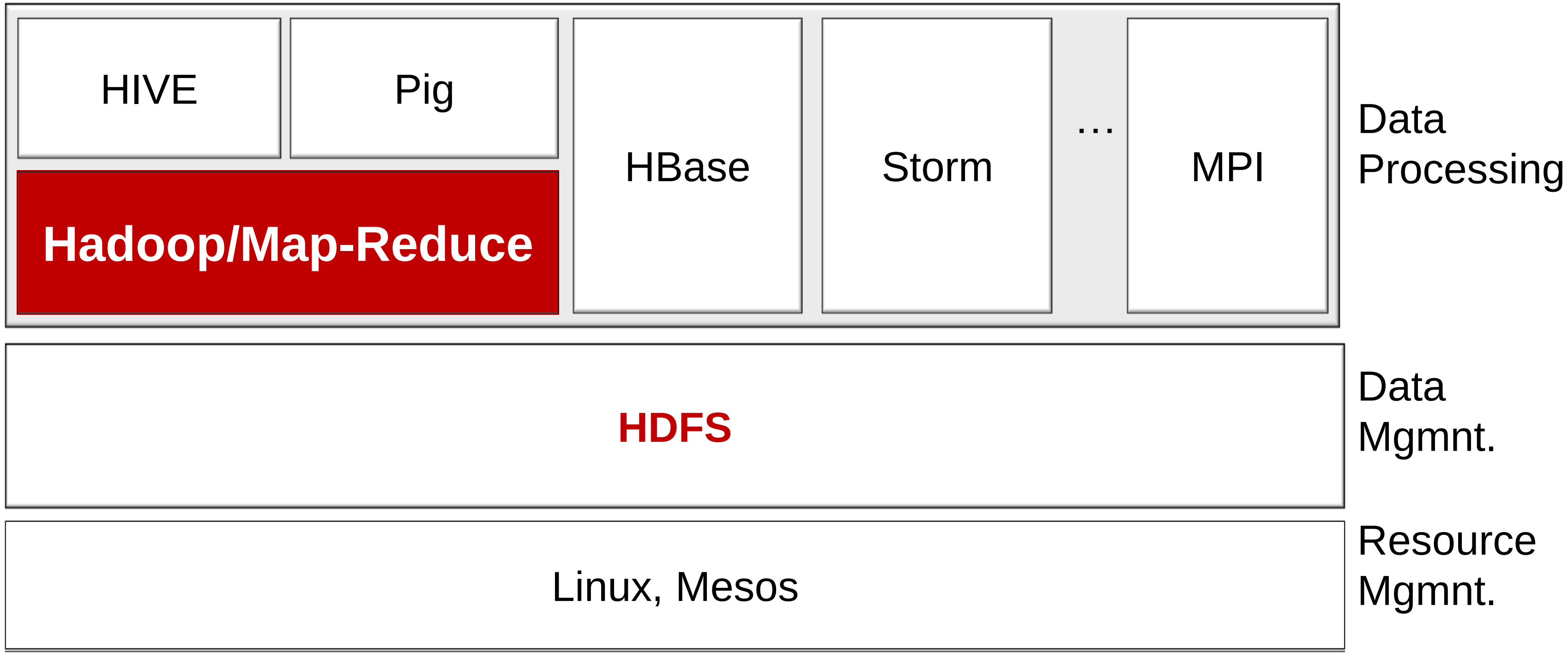
This year Google published a white paper describing the MapReduce framework, Doug Cutting and Mike Cafarella created Apache Hadoop **– later Big Data**



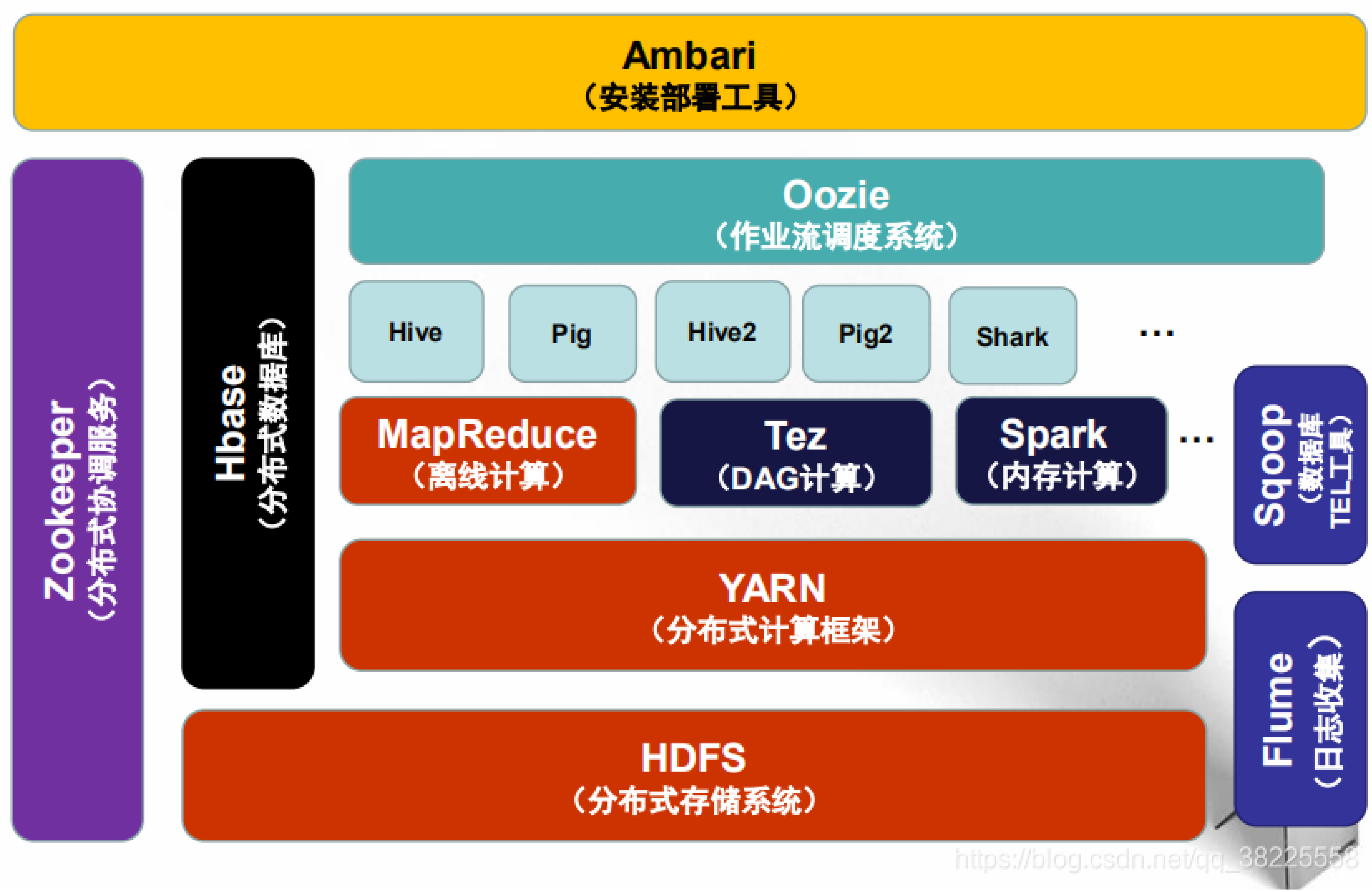
3 challenges for Big Data

- **How to store Big Data efficiently**
  - Divide and Conquer – distributed file system: **HDFS** (Hadoop Distributed File System) – based on **Linux**
    - Using Redundancy to support fault tolerance
- **How to provide friendly data manipulation?**
  - **Map/Reduce** is too naïve – keyword searching
  - HiveQL for HIVE **Data Warehouse**
  - SQL to support interaction with RDBMS – **Spark SQL**
- **How to integrate data analytics with Big Data?**
  - HIVE, Mahout
    - **Traditionally on M/R**
    - **Now on Spark**

1st generation of Big Data frameworks - Hadoop + M/R



- **The data processing is based on M/R model, which required intermediate result be kept into/out of disks – low performance**



Ambari
（安装部署工具）

Zookeeper
（分布式协调服务）

Hbase
（分布式数据库）

Oozie
（作业流调度系统）

Hive

Pig

Hive2

Pig2

Shark

…

MapReduce
（离线计算）

Tez
（DAG计算）

Spark
（内存计算）

…

Sqoop
（数据库TEL工具）

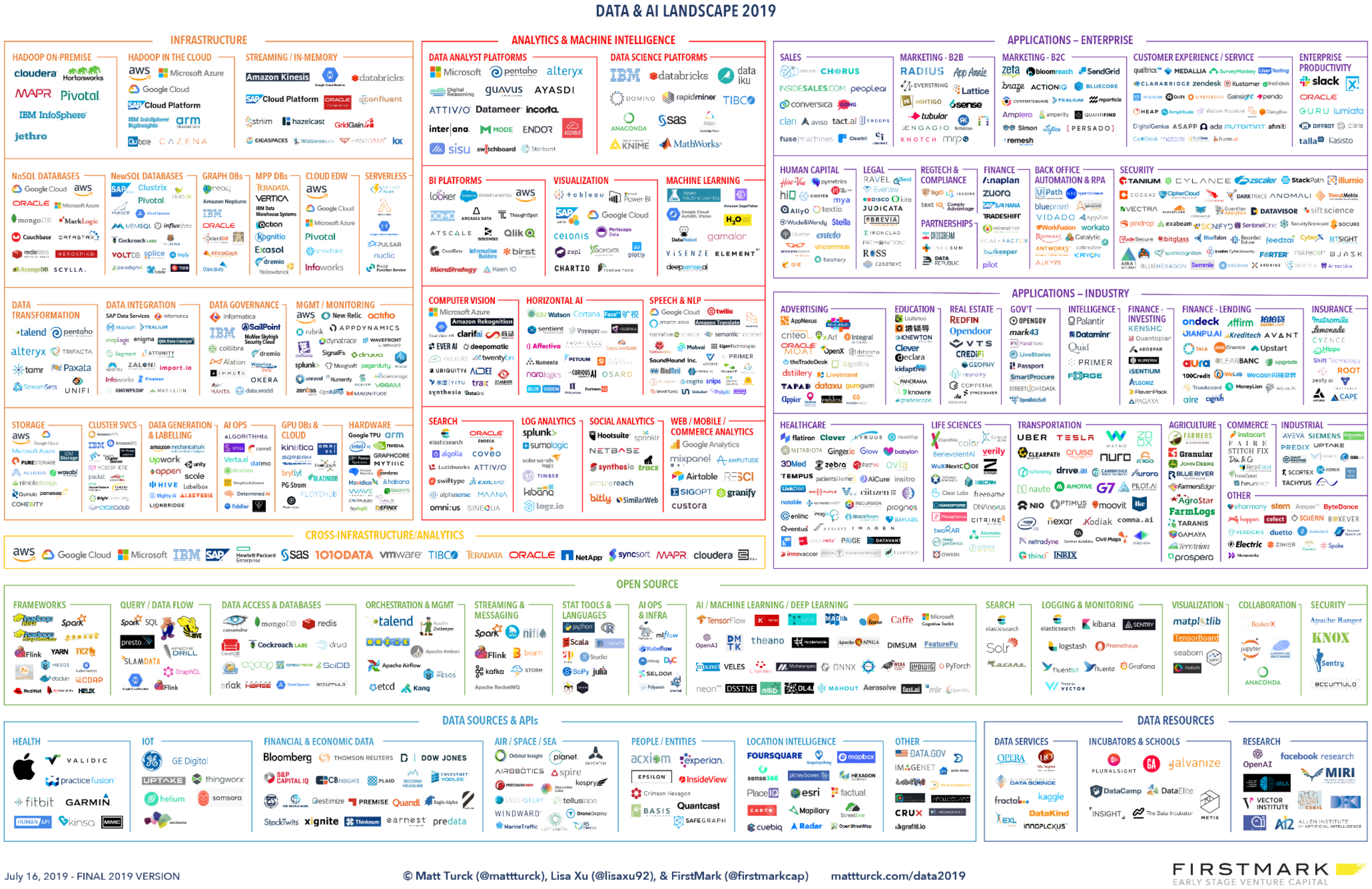
YARN
（分布式计算框架）

Flume
（日志收集）

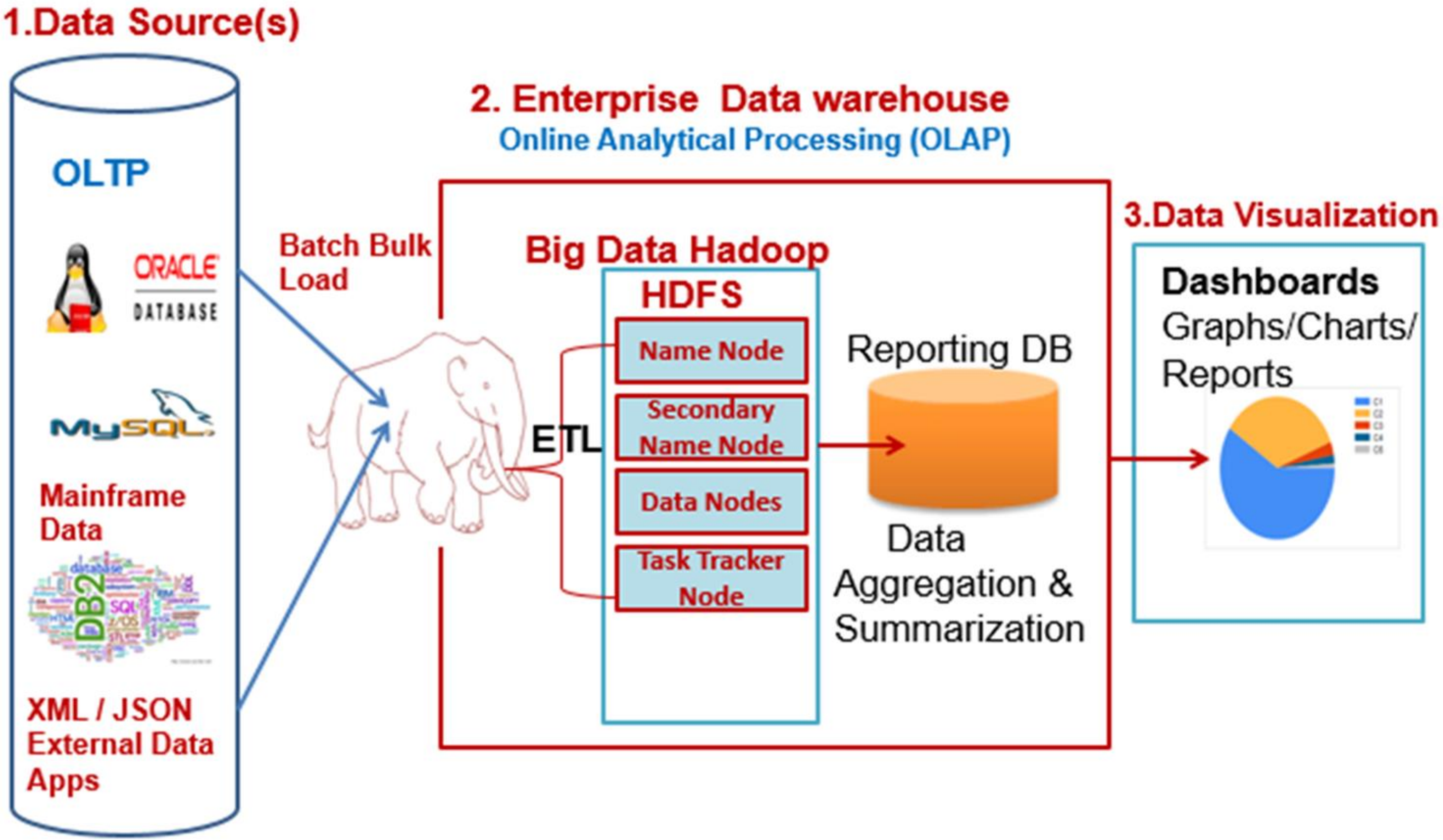
HDFS
（分布式存储系统）

https://zdnet4.cbsistatic.com/hub/i/2019/07/17/b17497a0-844e-42b7-9dde-9020902c0b46/32387d8af74af2ae02ed28429ba11dd0/big-data-landscape-2019-v7.png

The 2019 Big Data Landscape



Data Warehouse on Big Data



1.Data Source(s)

OLTP

Mainframe Data

XML / JSON External Data Apps

Batch Bulk Load

2. Enterprise Data warehouse

Online Analytical Processing (OLAP)

Big Data Hadoop

ETL

HDFS

Name Node

Secondary Name Node

Data Nodes

Task Tracker Node

Reporting DB

Data Aggregation & Summarization

3.Data Visualization

Dashboards

Graphs/Charts/ Reports

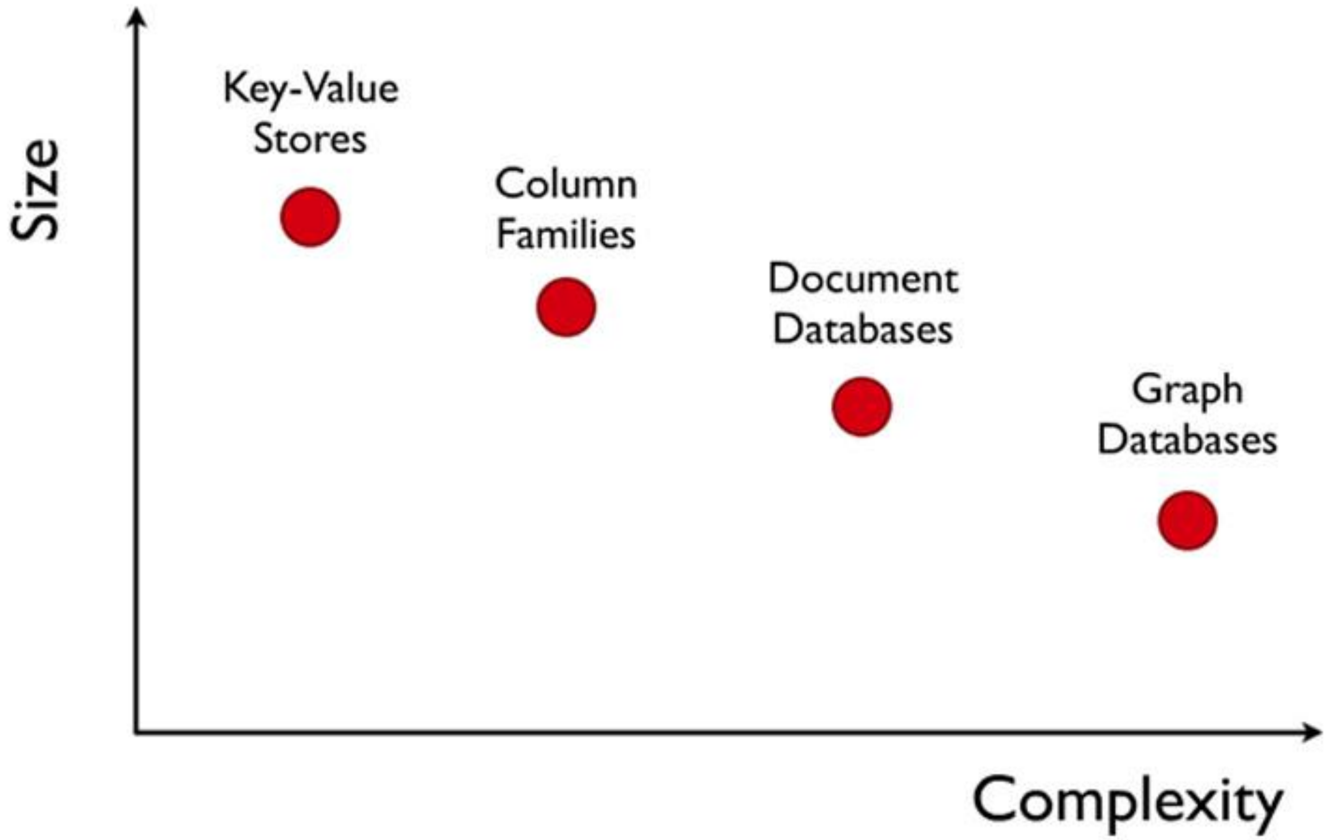
NoSQL?

- **Not Only SQL**
  - Hashing for fast data matching/updating

From www.nosql-database.org:

> **Next Generation Databases mostly addressing some of the points: being non-relational, distributed, open-source and horizontal scalable. The original intention has been modern web-scale databases. The movement began early 2009 and is growing rapidly. Often more characteristics apply as: schema-free, easy replication support, simple API, eventually consistent / BASE (not ACID), a huge data amount, and more.**

□ 4 Categories



- **Like the Shopping cart [购物篮]**
  - <key, values> is definitely more efficient for management –
    - insert, delete, updating, select
  - Than SQL

- **NoSQL Example #1: Key-Value Store**
  - **Hash tables of Keys**
  - Values stored with Keys
  - Example – Project-Voldemort
    - http://www.project-voldemort.com/ for Linkedin
  - Example – MemCacheDB
    - http://memcachedb.org/ Backend storage is Berkeley-DB

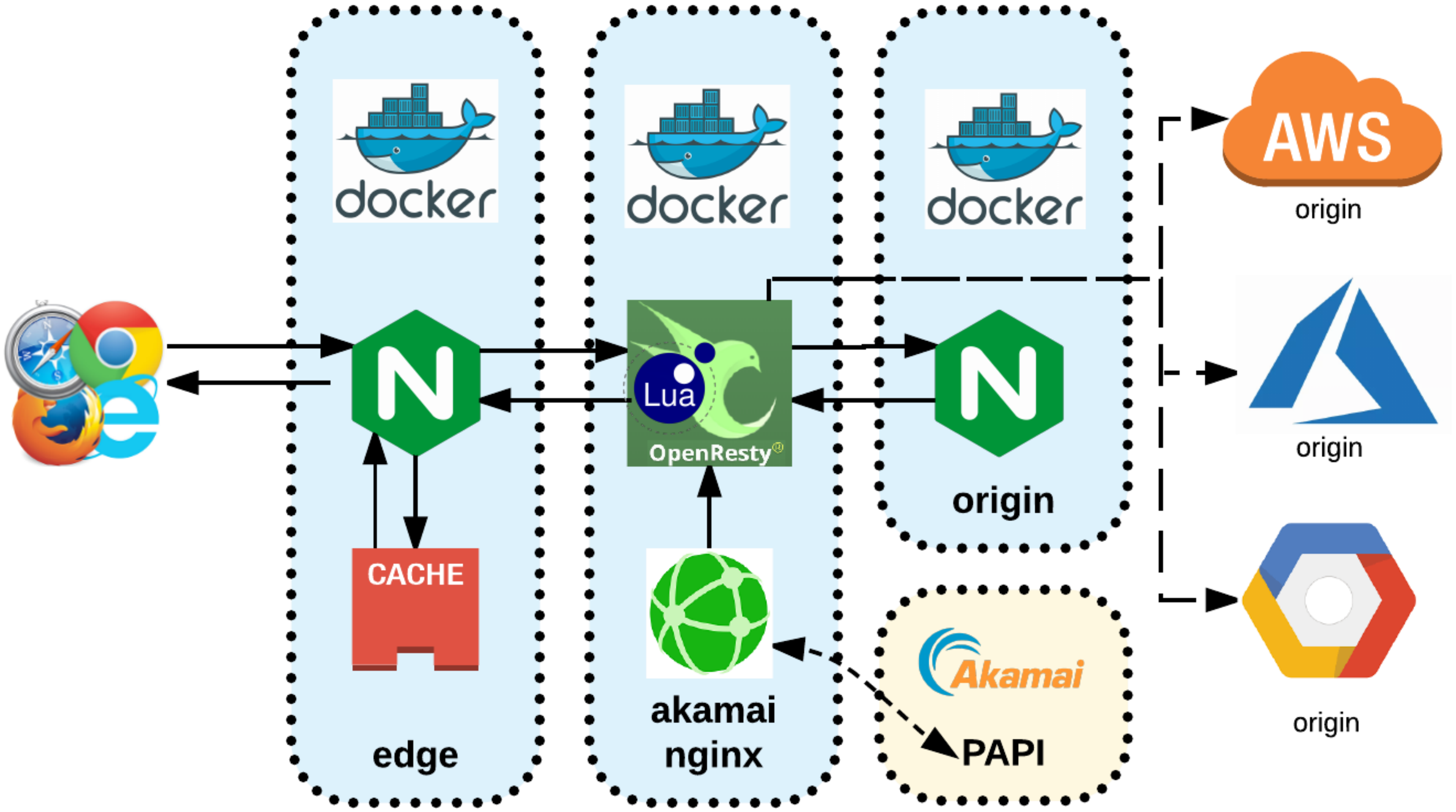
Example #2: CouchDB JSON Example

```
{
  "_id": "guid goes here",
  "_rev": "314159",

  "type": "abstract",

  "author": "Keith W. Hare"

  "title": "SQL Standard and NoSQL Databases",

  "body": "NoSQL databases (either no-SQL or Not Only SQL)
           are currently a hot topic in some parts of
           computing.",
  "creation_timestamp": "2011/05/10 13:30:00 +0004"
}
```

Clouding – a new trend

- Many Cloud platforms

akamai-nginx request flow

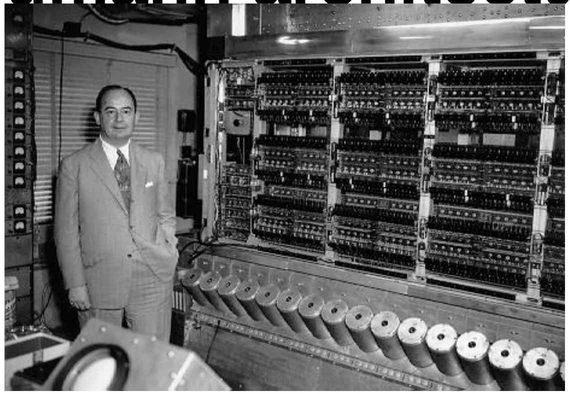


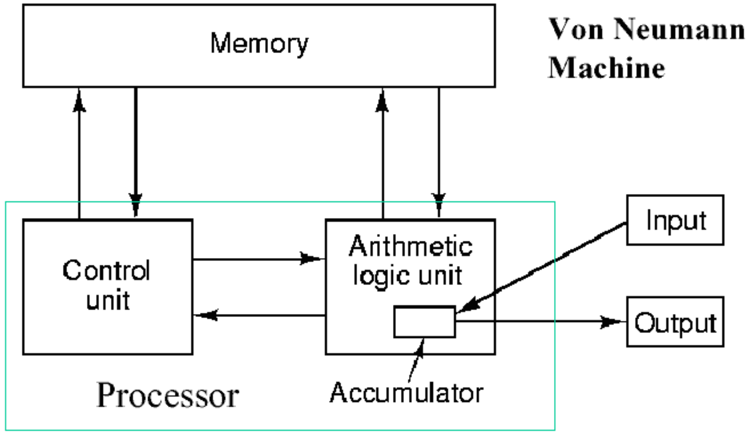
Akamai PAPI is called once, to generate equivalent LUA config

大商务，需要大数据

- **In IT age, platform sticking consumers [黏着客户的平台] is the popular pattern for e-business – Amazon, Google, Alibaba, JD, …**
  - **Large scale** (data and computing power) is important, which needs HPC
- **Large Scale Data – Big Data**
  - From File to Big Data
- **Large Scale computing power – High Performance Computing is now popular for business**
  - According to Top 500, MPP and Cluster
- **Additional bonus for Scientific Computing**
  - Weather forecasting

von Neumann architecture computer is not enough





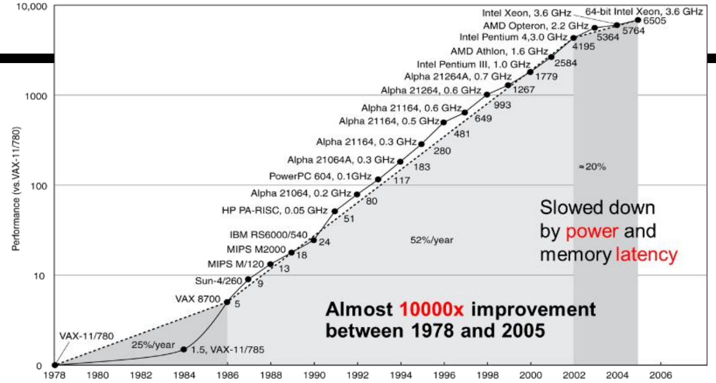
Walk-Through: *c=a+b*

1. Get next instruction
2. Decode: Fetch ***a***
3. Fetch ***a*** to internal register
4. Get next instruction
5. Decode: fetch ***b***
6. Fetch ***b*** to internal register
7. Get next instruction
8. Decode: add ***a*** and ***b*** (***c*** in register)
9. Do the addition in ALU
10. Get next instruction
11. Decode: store ***c*** in main memory
12. Move ***c*** from internal register to main memory

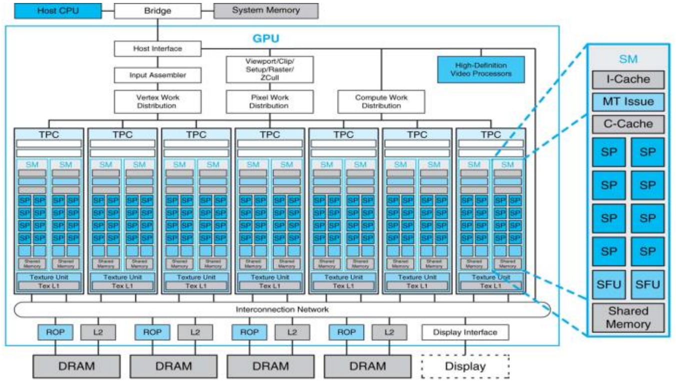
Note: Some units are idle while others are working…waste of cycles.
Pipelining (modularization) & Cashing (advance decoding)…parallelism

3 ideas to get powerful computing

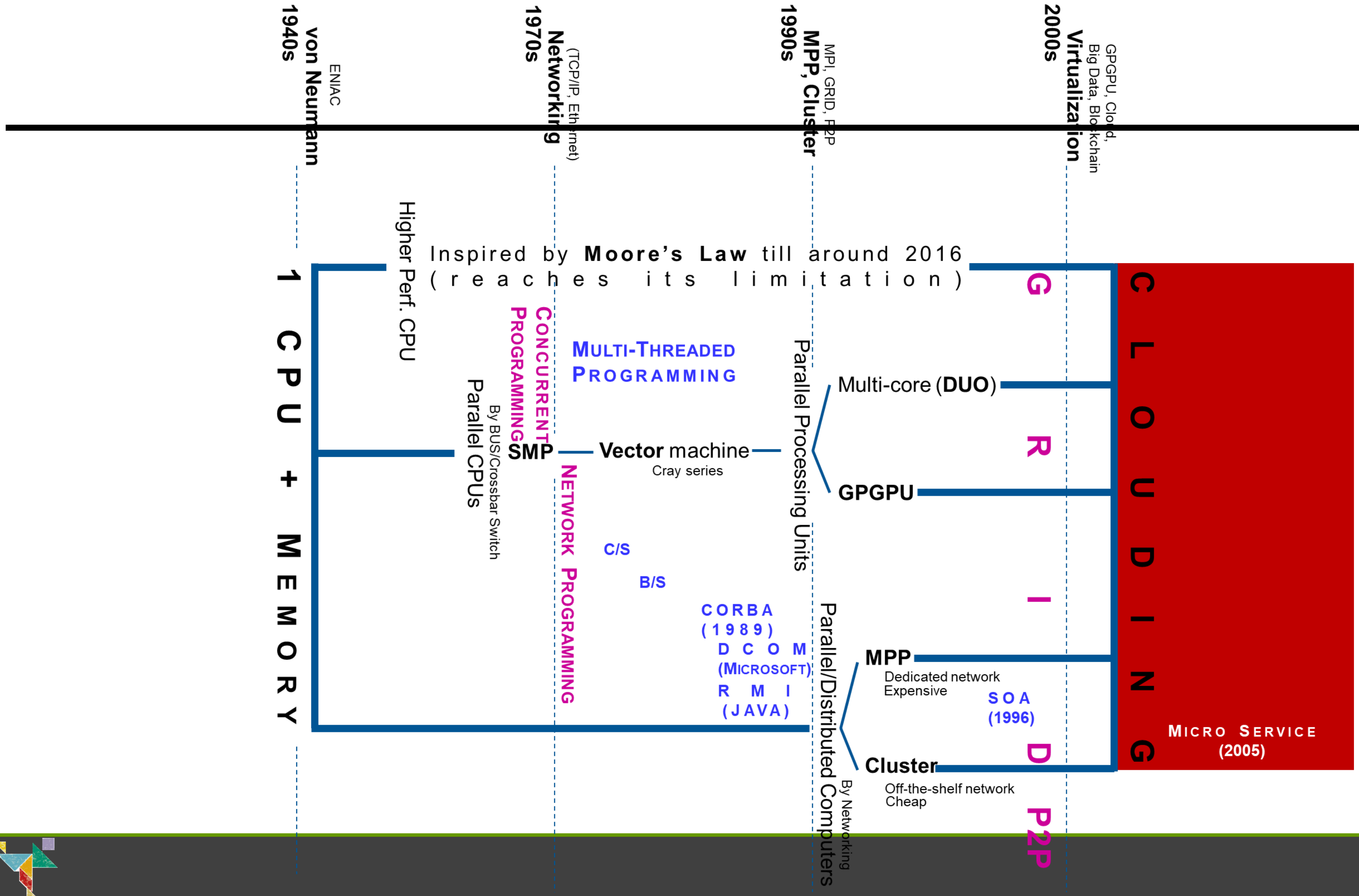
- **Integrate more circuits in one processor (CPU)**
  - Limitation – **Moore’s law**



- **Integrate more processing in one computer – Parallel (MPP)**
  - Limitation – storage and CPUs
  - **GPU** is a good idea



- **Integrate more computers to work together – Distributed (Cluster)**
  - Limitation – connection speed (Special bus or network)



1940s
von Neumann
ENIAC

1970s
Networking
(TCP/IP, Ethernet)

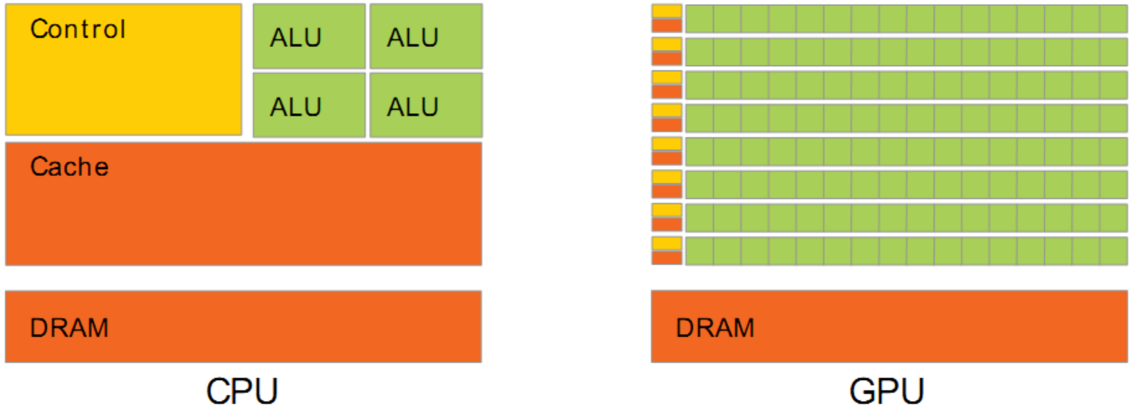
1990s
MPP, Cluster
MPI, GRID, P2P

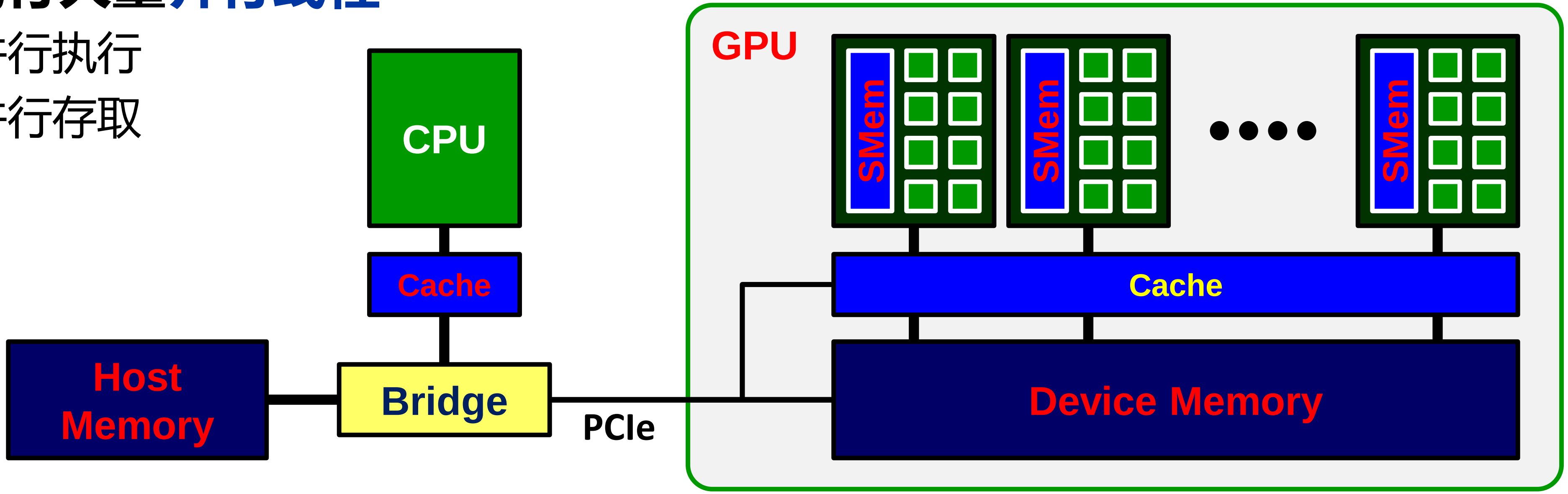
2000s
Virtualization
GPGPU, Cloud, Big Data, Blockchain

Inspired by **Moore's Law** till around 2016 (reaches its limitation)

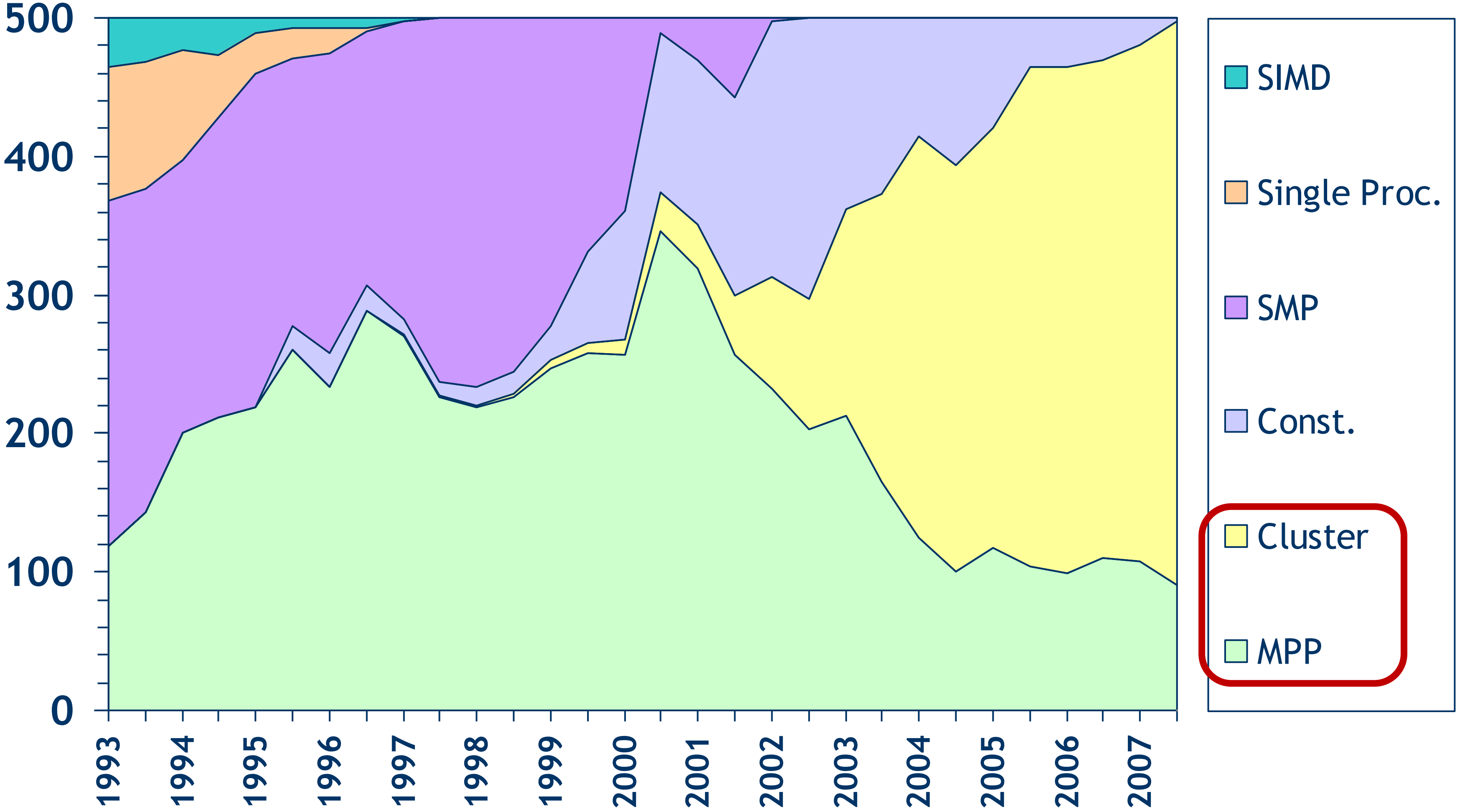
Using CPU+GPU Architecture

- □ **CPU+GPU异构多核系统**
  - 针对每个任务选择合适的处理器和存储器
- □ **通用CPU 适合执行一些串行的线程**
  - 串行执行快
  - 带有cache，访问存储器延时低
- □ **GPU 适合执行大量并行线程**
  - 可扩放的并行执行
  - 高带宽的并行存取



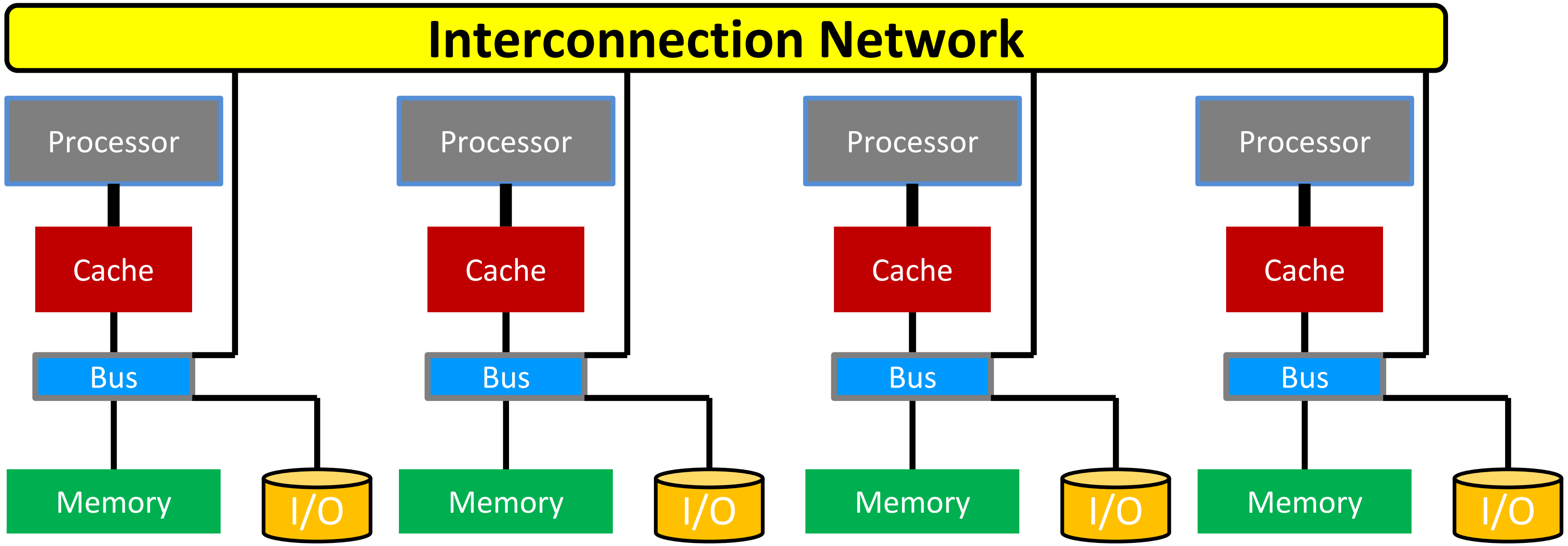


The 30th List as of November 2007



MPP: Massively Parallel Processors

- Massively Parallel **Processors** (MPP) architecture consists of nodes with each having its own processor, memory and I/O subsystem
- An independent OS runs at each node



NASA Beowulf Project – 1994

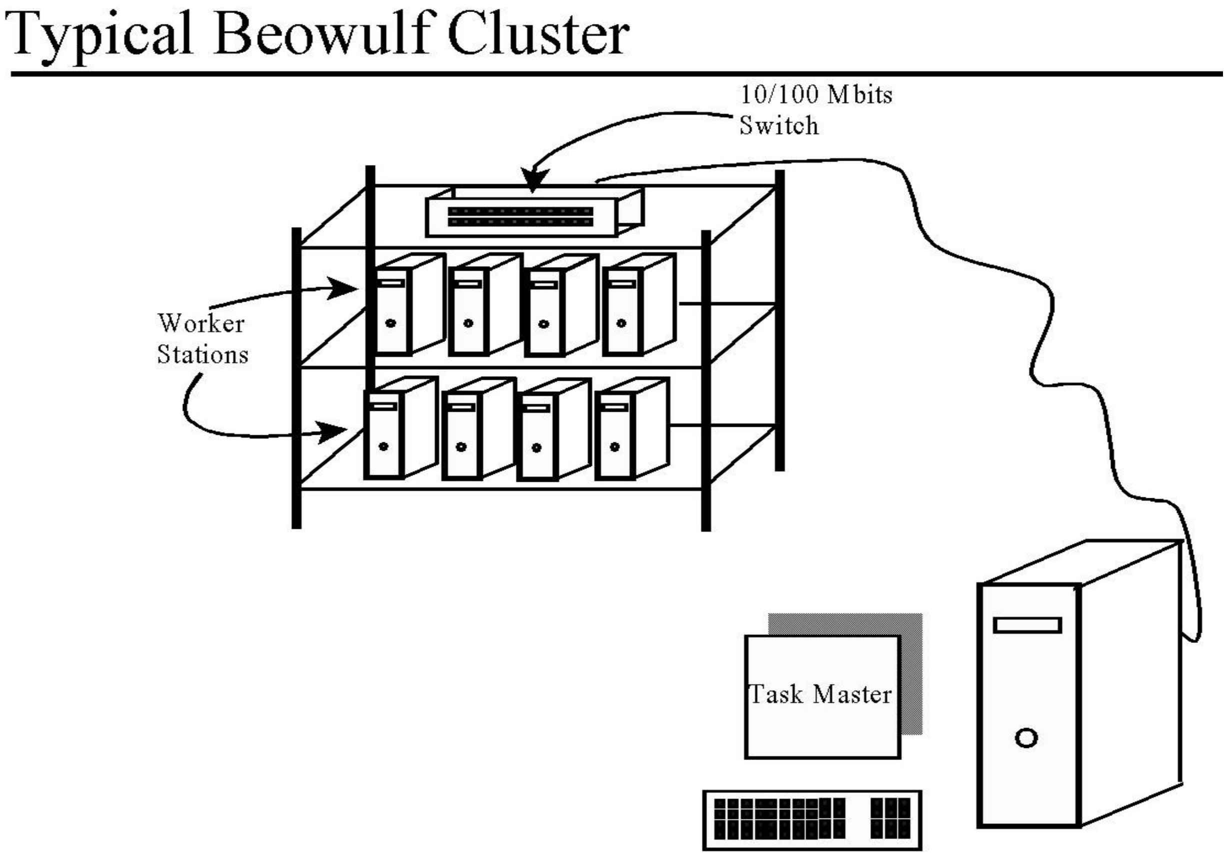
- Wiglaf - 1994
- 16 Intel 80486 100 MHz
- VESA Local bus
- 256 Mbytes memory
- 6.4 Gbytes of disk
- Dual 10 base-T Ethernet
- 72 Mflops sustained
- $40K

- Hrothgar - 1995
- 16 Intel Pentium100 MHz
- PCI
- 1 Gbyte memory
- 6.4 Gbytes of disk
- 100 base-T Fast Ethernet (hub)
- 240 Mflops sustained
- $46K

- Hyglac-1996 (Caltech)
- 16 Pentium Pro 200 MHz
- PCI
- 2 Gbytes memory
- 49.6 Gbytes of disk
- 100 base-T Fast Ethernet (switch)
- 1.25 Gflops sustained
- $50K

Beowulf Cluster Architecture

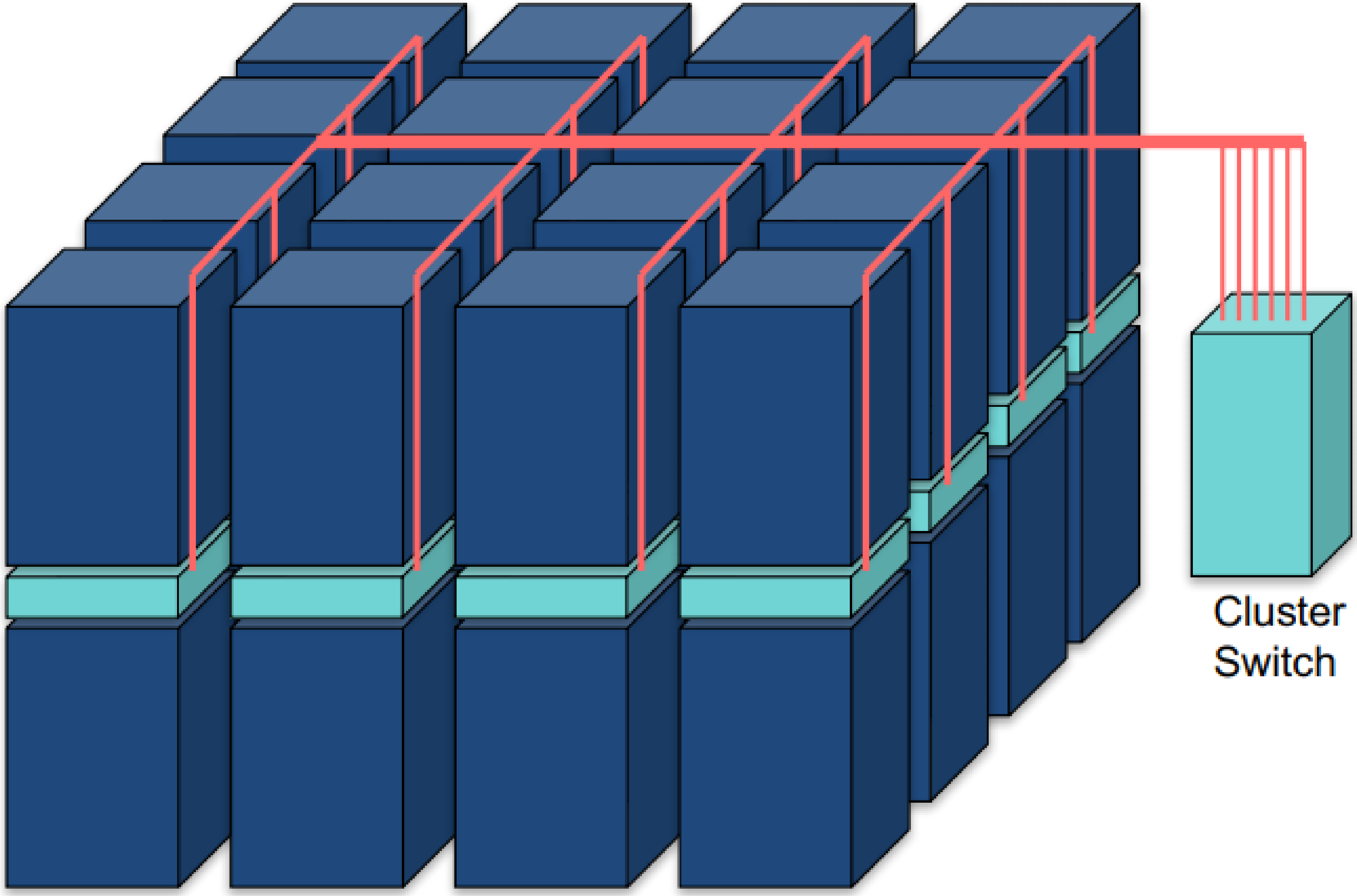
- **Master-Slave configuration**
- **Master Node**
  - Job scheduling
  - System monitoring
  - Resource management
- **Slave Node**
  - Does assigned work
  - Communicates with other slave nodes
  - Sends results to master node



Backrub (Google) 1997

The cluster now – Data Center

https://www.cs.rutgers.edu/~pxk/417/notes/content/22-clusters-slides.pdf



Cluster of 4×4 racks

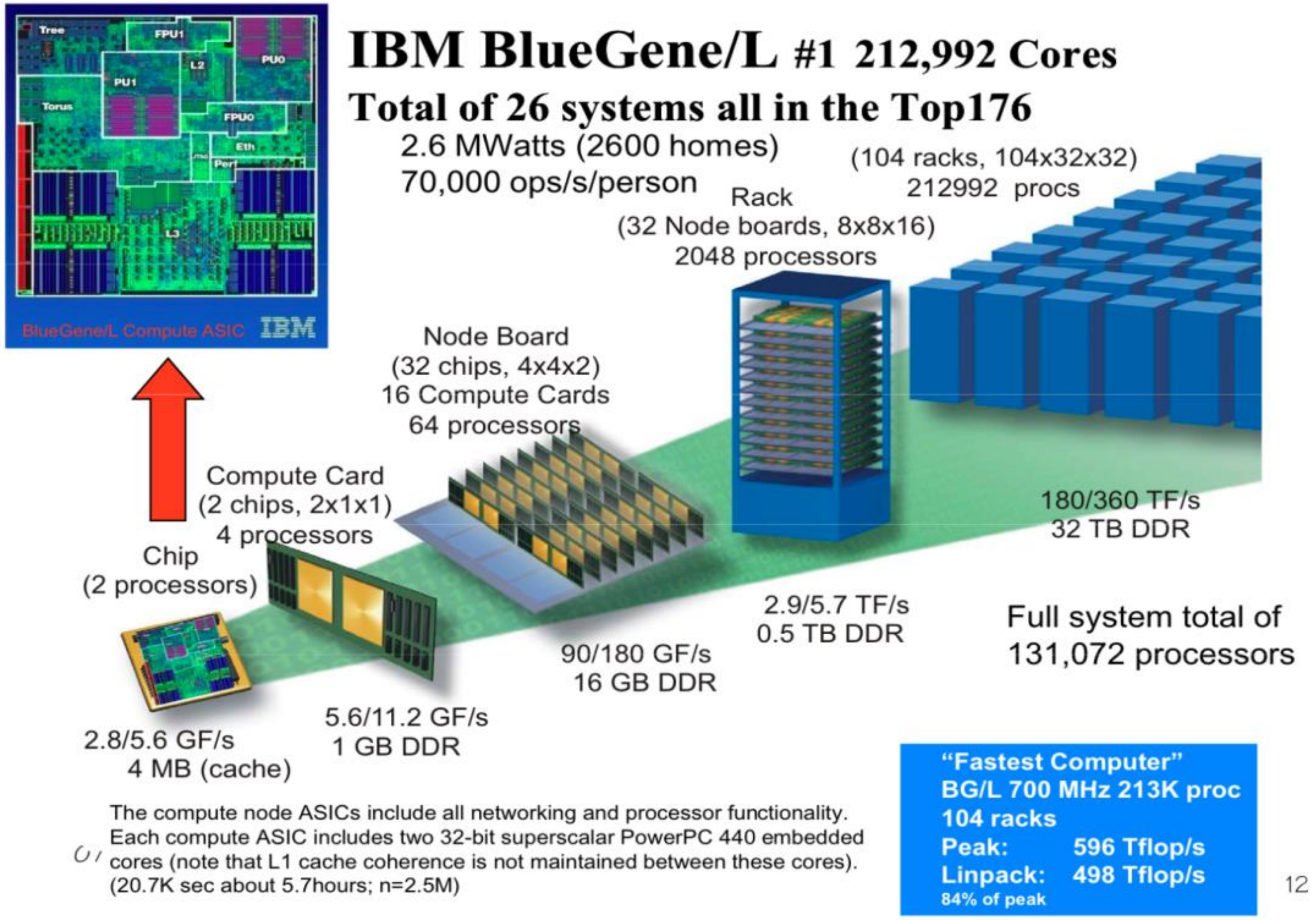
Assume:
10 Gbps per server
40 servers per rack
⇒ 400 Gbps/rack

16 racks
⇒ 8 Tbps

Max switch capacity currently ~ 5 Tbps
⇒ Need at least two cluster switches

Cluster is also the popular architecture for SuperComputer

- IBM, 2004. BlueGene/L
- First supercomputer ever to run over 100 TFLOPS sustained on a real world application, namely a three-dimensional molecular dynamics code (ddcMD).



IBM BlueGene/L #1 212,992 Cores

Total of 26 systems all in the Top176

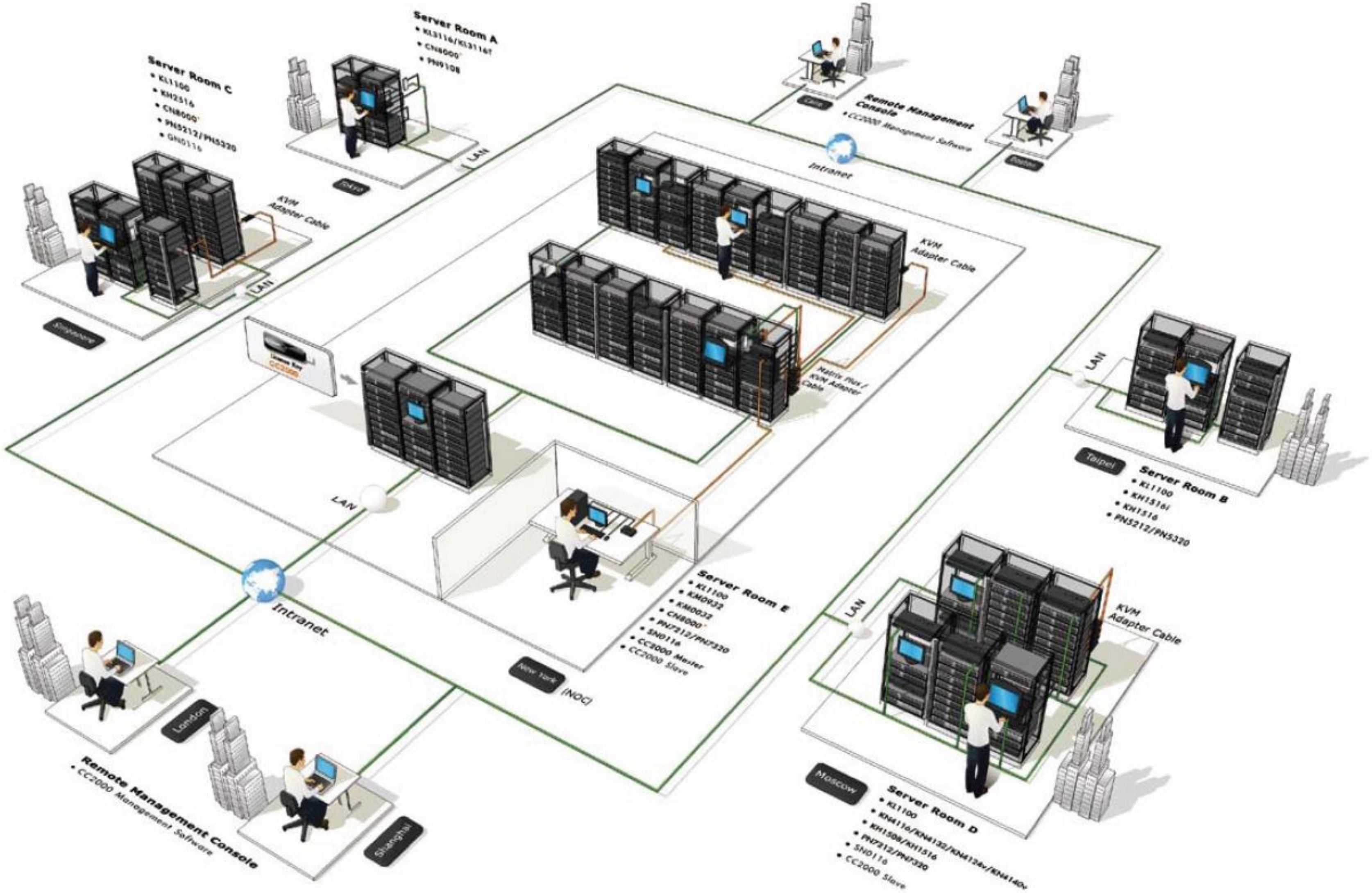
2.6 MWatts (2600 homes)
70,000 ops/s/person

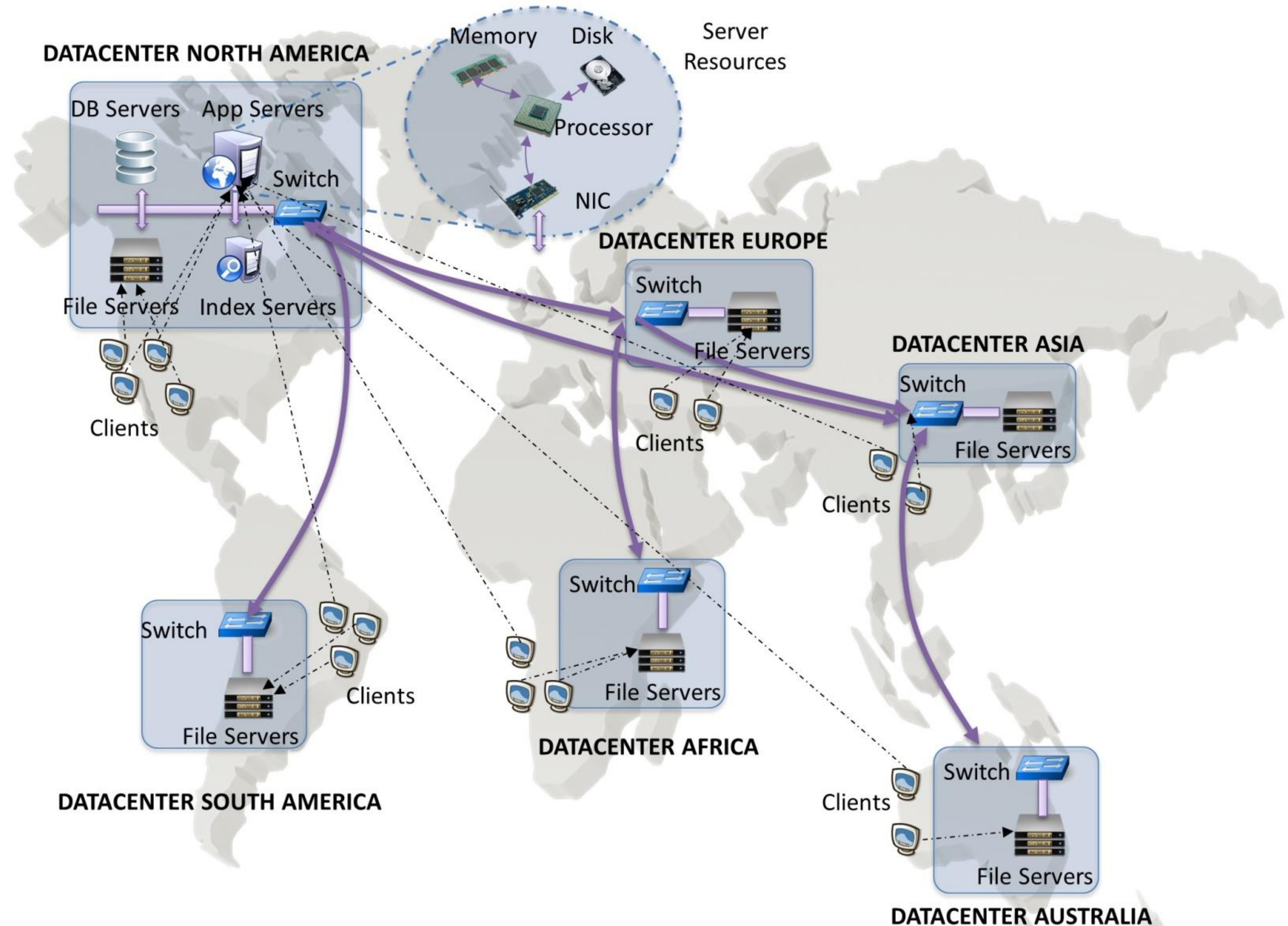
The compute node ASICs include all networking and processor functionality. Each compute ASIC includes two 32-bit superscalar PowerPC 440 embedded cores (note that L1 cache coherence is not maintained between these cores). (20.7K sec about 5.7hours; n=2.5M)

"Fastest Computer"
BG/L 700 MHz 213K proc
104 racks
Peak: 596 Tflop/s
Linpack: 498 Tflop/s
84% of peak

12

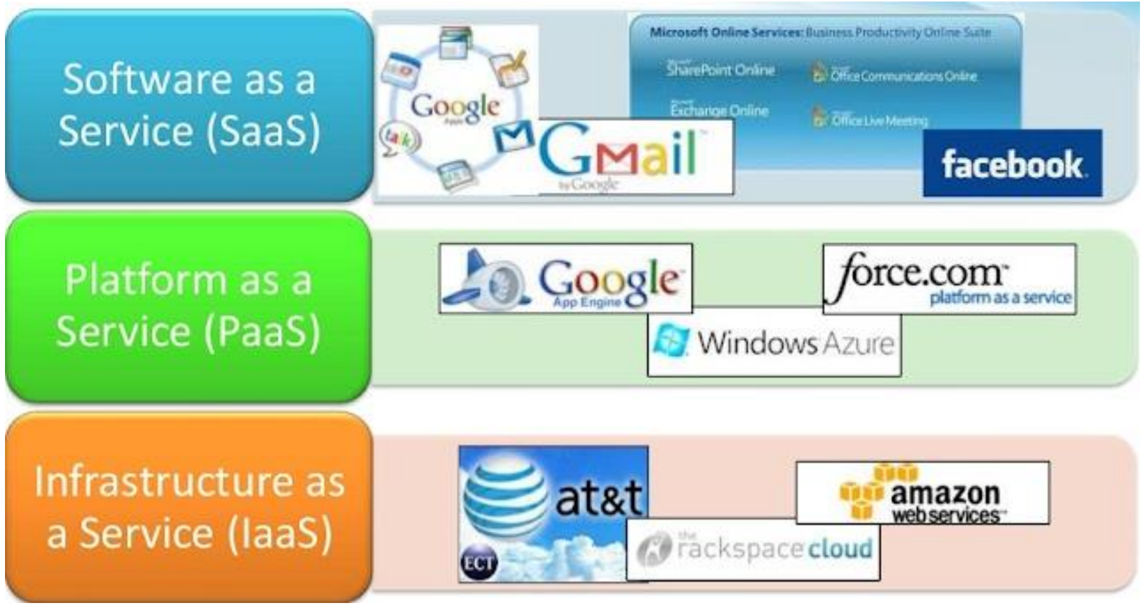
Data Center is a “computer”



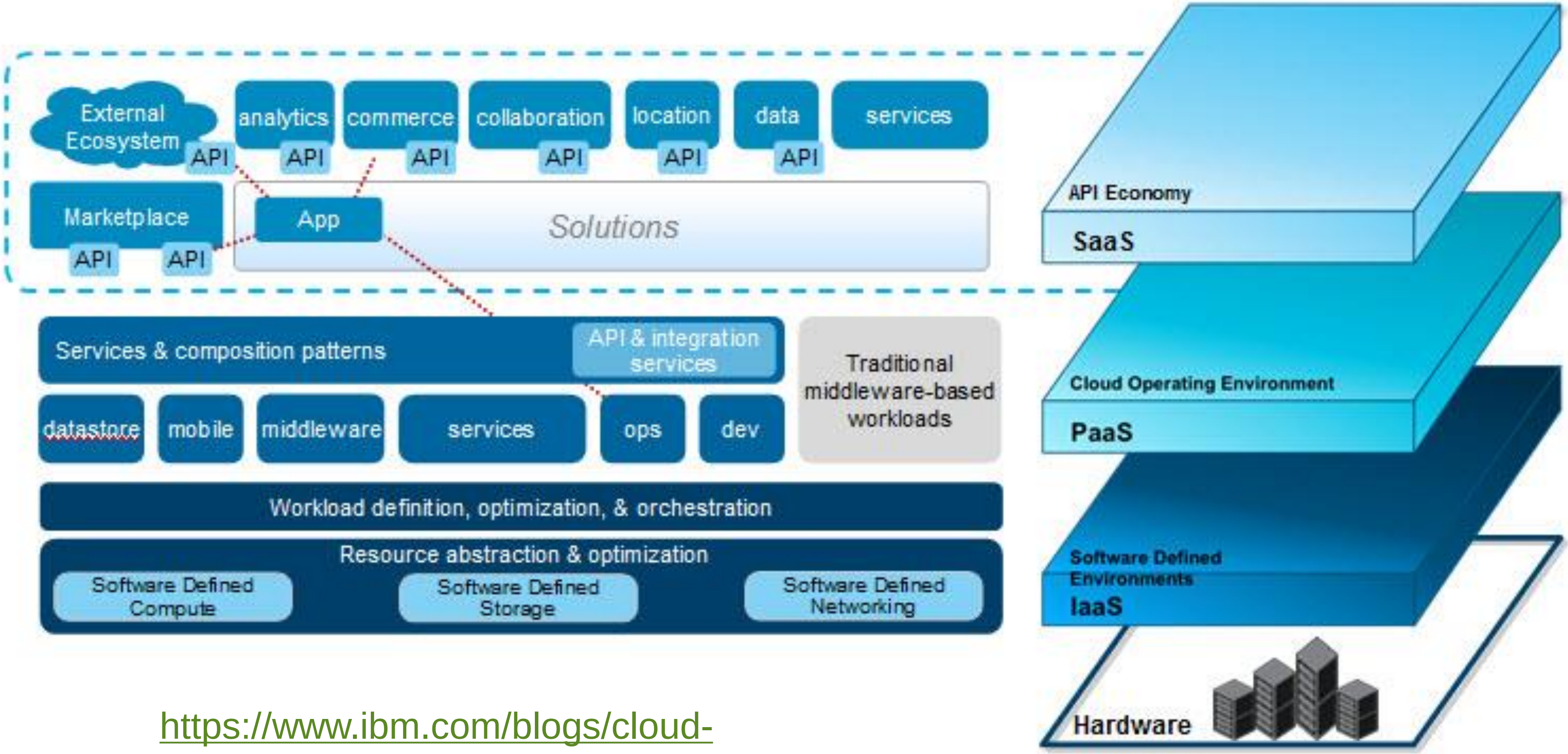

DATACENTER NORTH AMERICA
DB Servers
App Servers
Switch
File Servers
Index Servers
Clients
Memory
Disk
Processor
NIC
Server Resources
DATACENTER EUROPE
Switch
File Servers
Clients
DATACENTER ASIA
Switch
File Servers
Clients
DATACENTER AFRICA
Switch
File Servers
DATACENTER SOUTH AMERICA
Switch
File Servers
Clients
DATACENTER AUSTRALIA
Switch
File Servers
Clients

Cloud – 2006

- **Originated in the business domain**
  - Outsourcing services; Pay for what you use
- **Provided by data centers built on computer and storage virtualization technologies.**

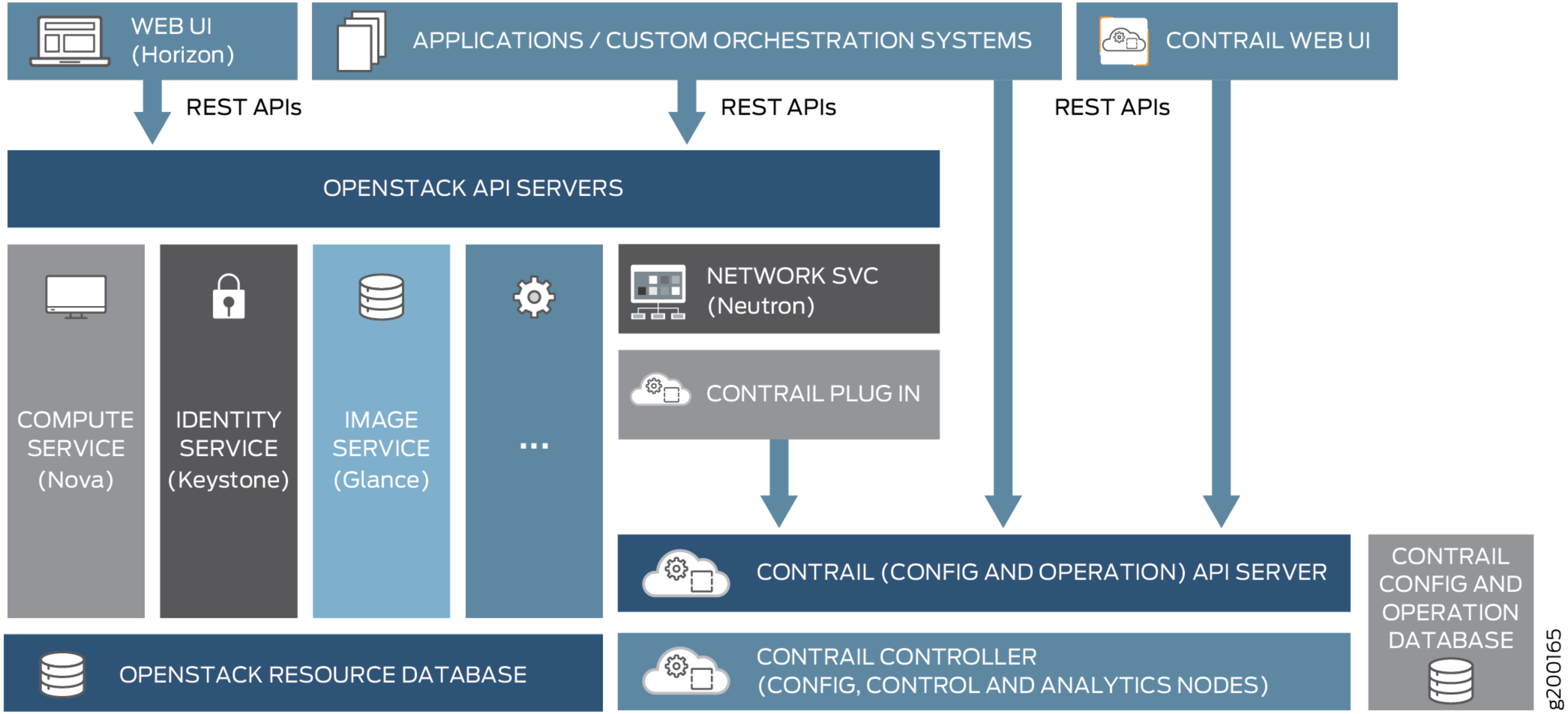

Software as a Service (SaaS)
Google
Gmail
Microsoft Online Services: Business Productivity Online Suite
SharePoint Online
Office Communications Online
Exchange Online
Office Live Meeting
facebook
Platform as a Service (PaaS)
Google App Engine
force.com platform as a service
Windows Azure
Infrastructure as a Service (IaaS)
at&t
amazon webservices
the rackspace cloud

- **Scientific applications often have different requirements**
  - MPI, Shared file system, Support for many dependent jobs


External Ecosystem
API
analytics
API
commerce
API
collaboration
API
location
API
data
API
services
Marketplace
API
API
App
Solutions
Services & composition patterns
API & integration services
datastore
mobile
middleware
services
ops
dev
Traditional middleware-based workloads
Workload definition, optimization, & orchestration
Resource abstraction & optimization
Software Defined Compute
Software Defined Storage
Software Defined Networking
API Economy
SaaS
Cloud Operating Environment
PaaS
Software Defined Environments
IaaS
Hardware

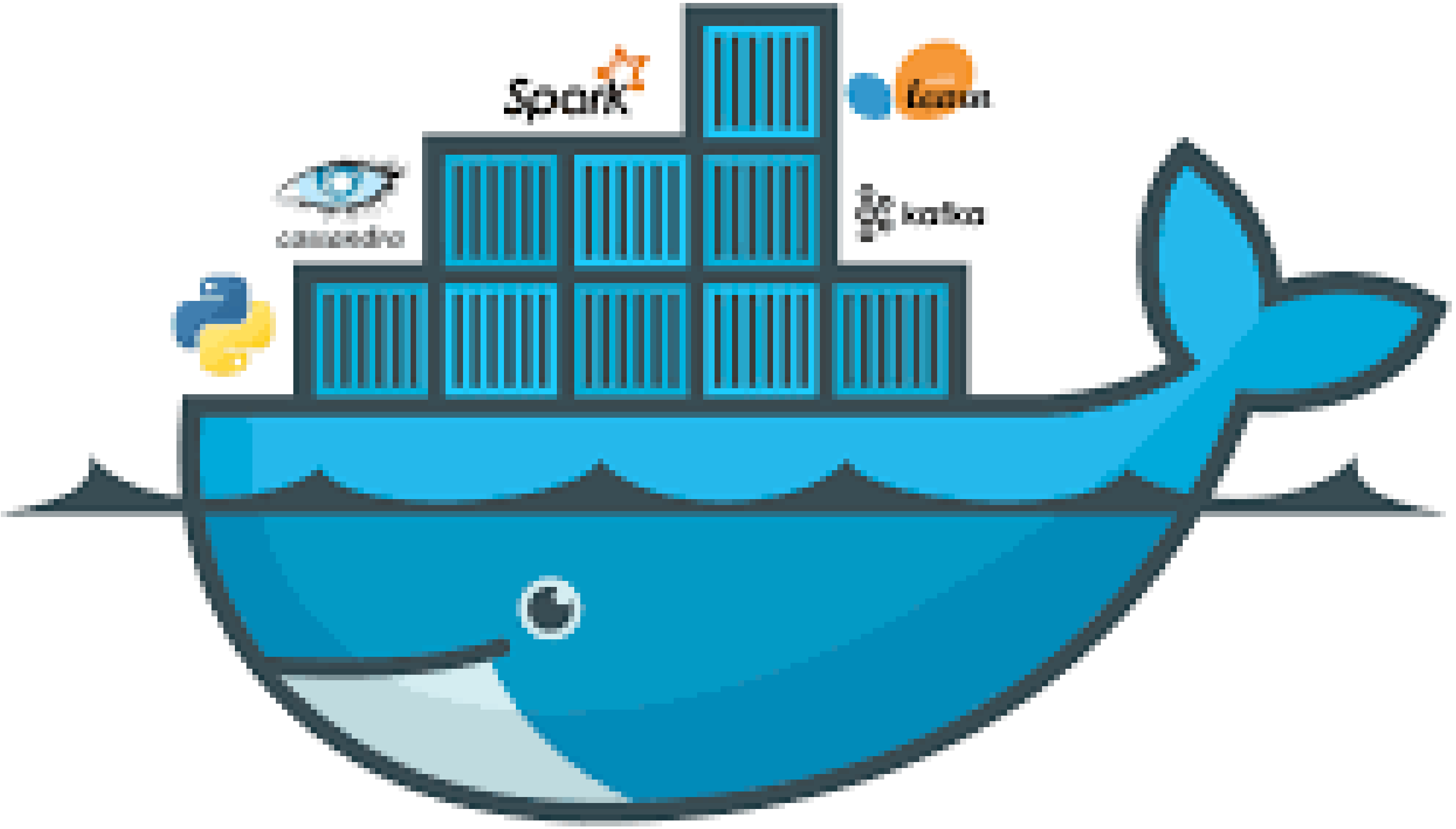
https://www.ibm.com/blogs/cloud-computing/2013/08/07/how-openpower-consortium-will-help-shape-the-open-cloud/

OpenStack

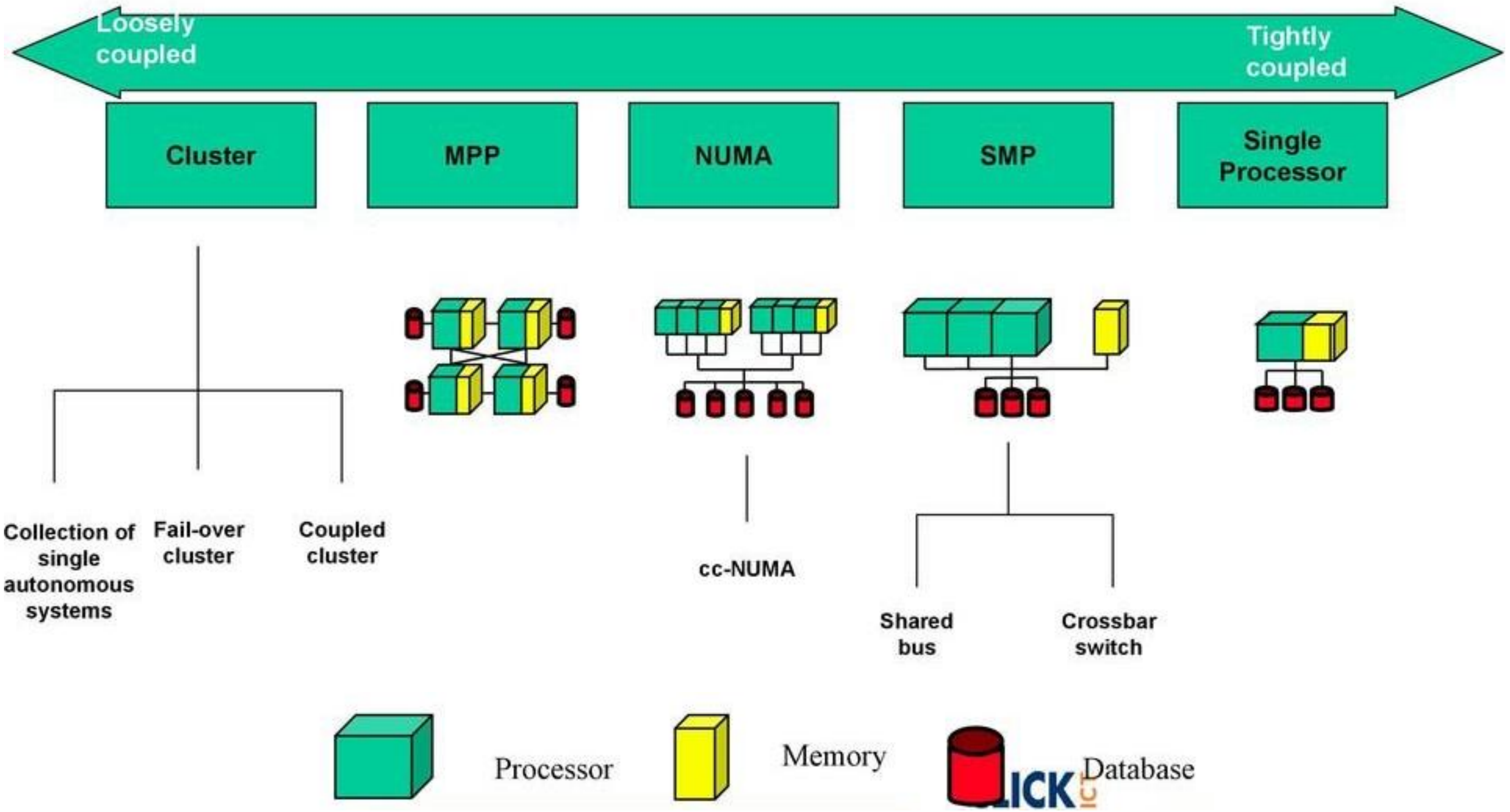


WEB UI
(Horizon)
APPLICATIONS / CUSTOM ORCHESTRATION SYSTEMS
CONTRAIL WEB UI
REST APIs
REST APIs
REST APIs
OPENSTACK API SERVERS
COMPUTE SERVICE (Nova)
IDENTITY SERVICE (Keystone)
IMAGE SERVICE (Glance)
...
NETWORK SVC
(Neutron)
CONTRAIL PLUG IN
CONTRAIL (CONFIG AND OPERATION) API SERVER
CONTRAIL CONFIG AND OPERATION DATABASE
OPENSTACK RESOURCE DATABASE
CONTRAIL CONTROLLER
(CONFIG, CONTROL AND ANALYTICS NODES)
g200165

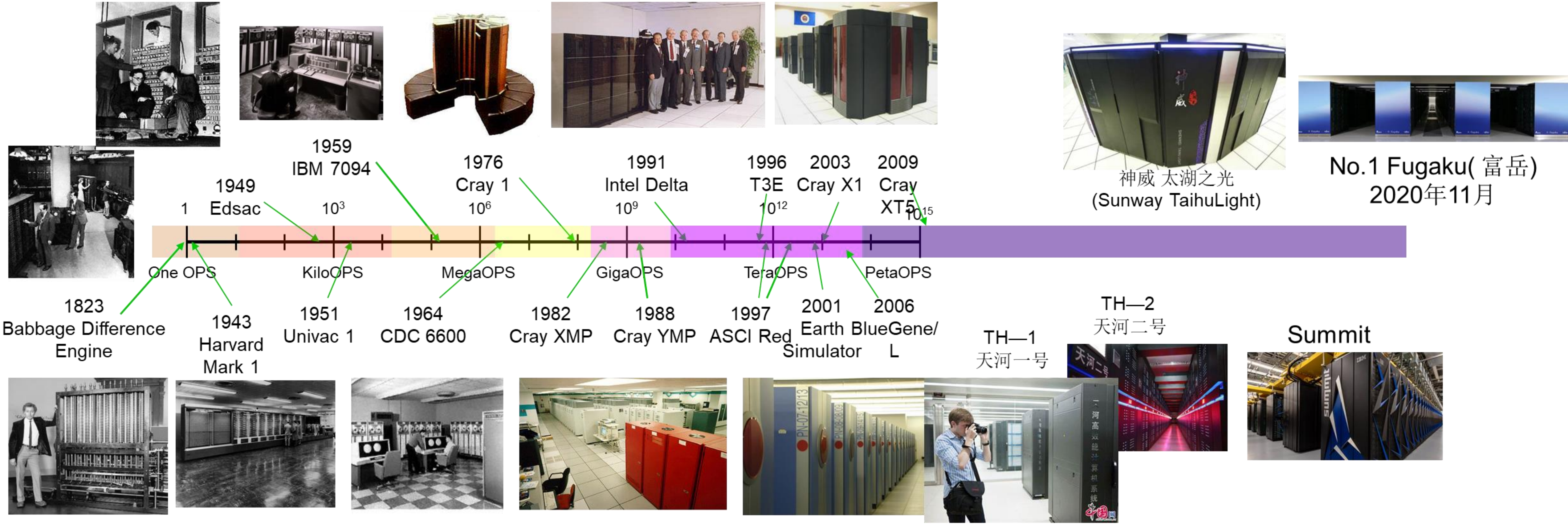
Containers



Multiprocessor system architectures



HPC - High Performance Computers



1 | 10³ | 10⁶ | 10⁹ | 10¹² | 10¹⁵

One OPS | KiloOPS | MegaOPS | GigaOPS | TeraOPS | PetaOPS

1823
Babbage Difference Engine

1943
Harvard Mark 1

1949
Edsac

1951
Univac 1

1959
IBM 7094

1964
CDC 6600

1976
Cray 1

1982
Cray XMP

1988
Cray YMP

1991
Intel Delta

1996
T3E

1997
ASCI Red

2001
Earth Simulator

2003
Cray X1

2006
BlueGene/L

2009
Cray XT5

TH—1
天河一号

TH—2
天河二号

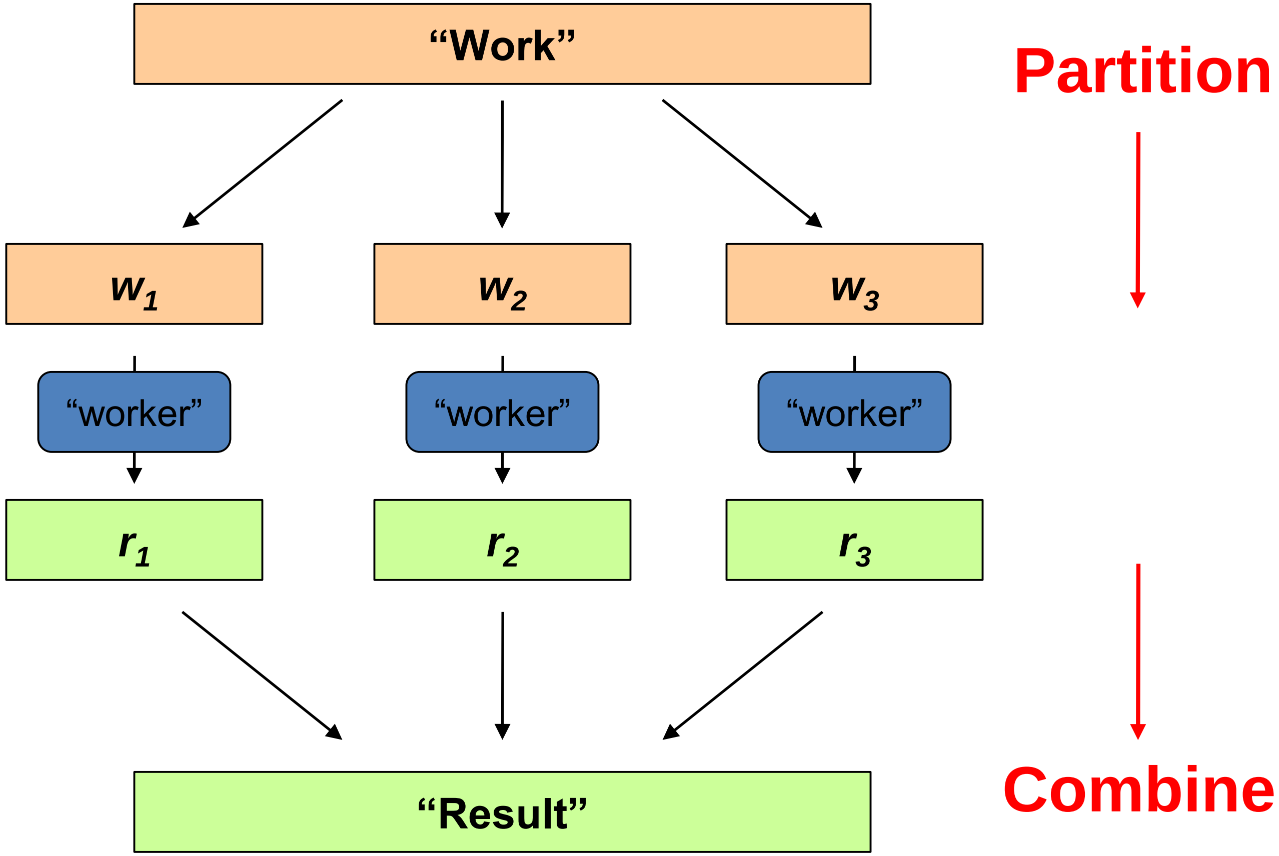
神威 太湖之光
(Sunway TaihuLight)

Summit

No.1 Fugaku(富岳)
2020年11月

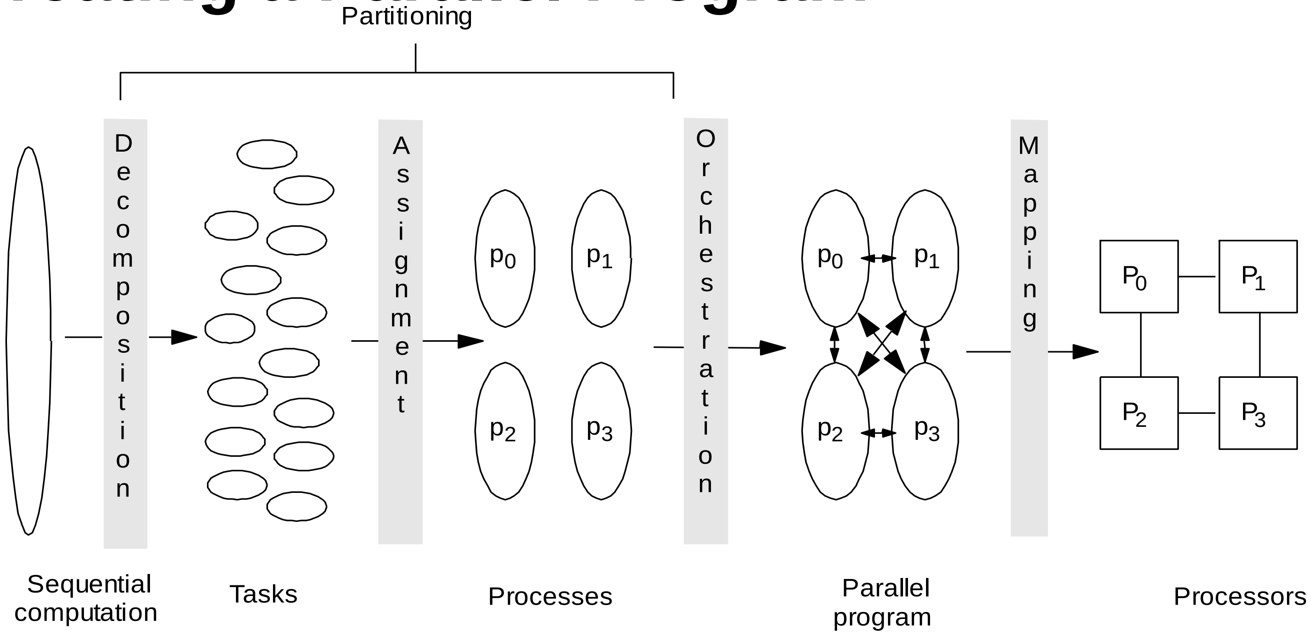
OPS: Operations Per Second **FLOPS**: Floating Point OPS

HPComputing: methodology for distributed/parallel programming? – Divide & Conquer



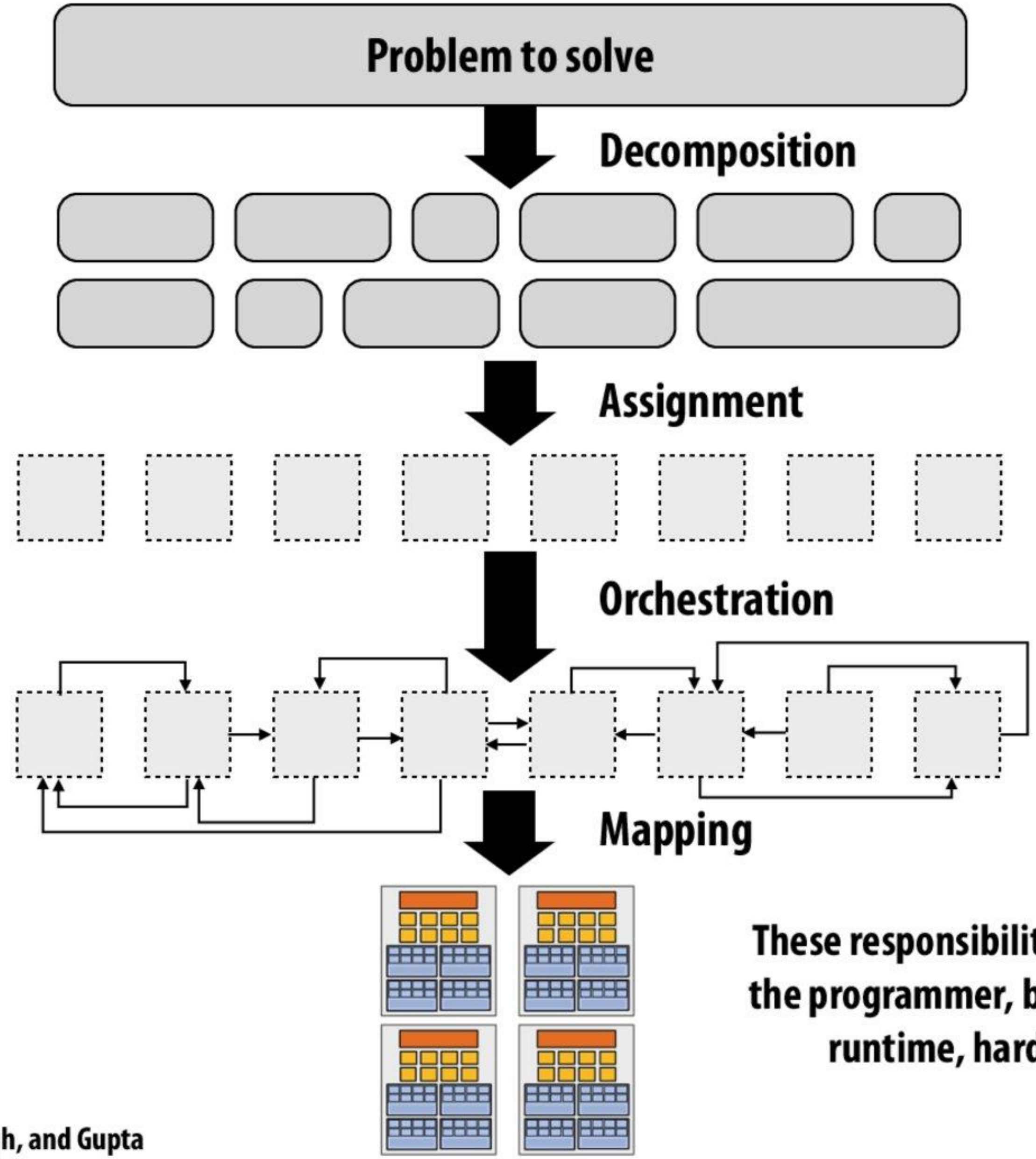
Ideas to convert Sequential to Parallel

4 Steps in Creating a Parallel Program



- **D**ecomposition of computation in tasks
- **A**ssignment of tasks to processes
- **O**rchestration of data access, comm, synch.
- **M**apping processes to processors

Creating a parallel program



Problem to solve

Decomposition

Subproblems
(a.k.a. “tasks”,
“work to do”)

Assignment

Parallel Threads **
(“workers”)

** I had to pick a term

Orchestration

Parallel program
(communicating
threads)

Mapping

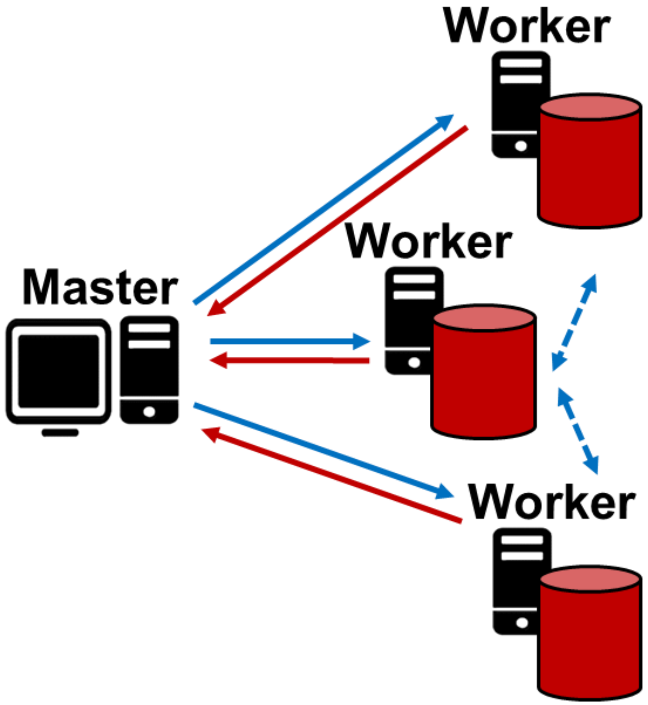
Execution on
parallel machine

These responsibilities may be assumed by the programmer, by the system (compiler, runtime, hardware), or by both!

Adopted from: Culler, Singh, and Gupta

□ 2 main architectures in distributed computing

Master-Slave



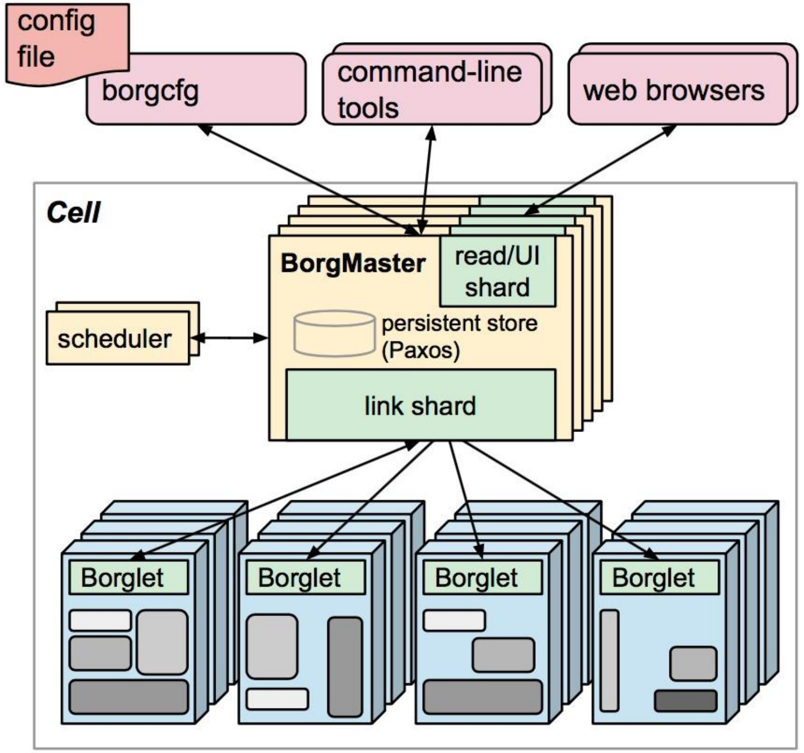
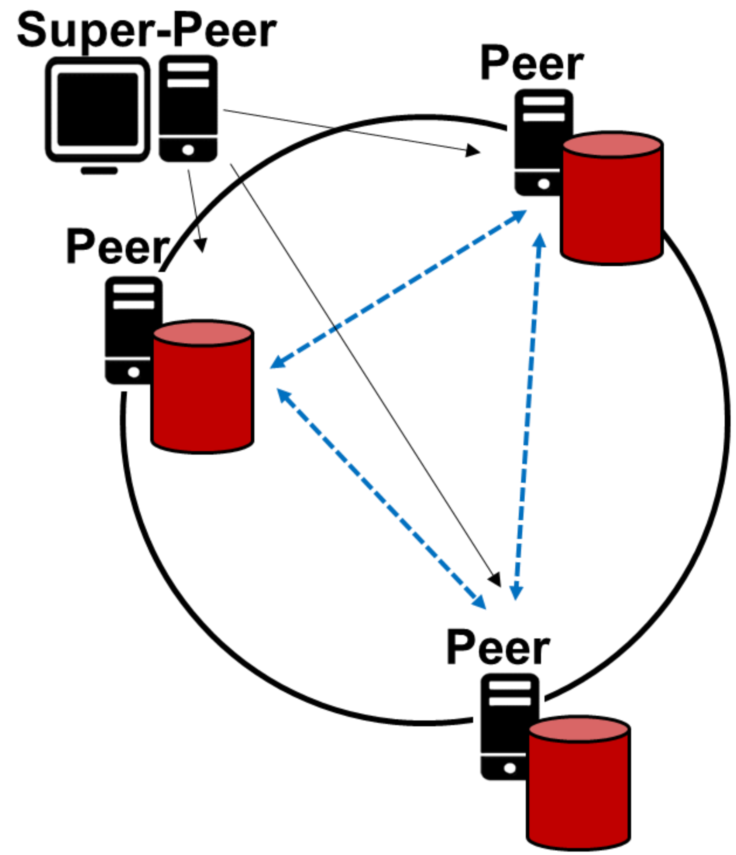
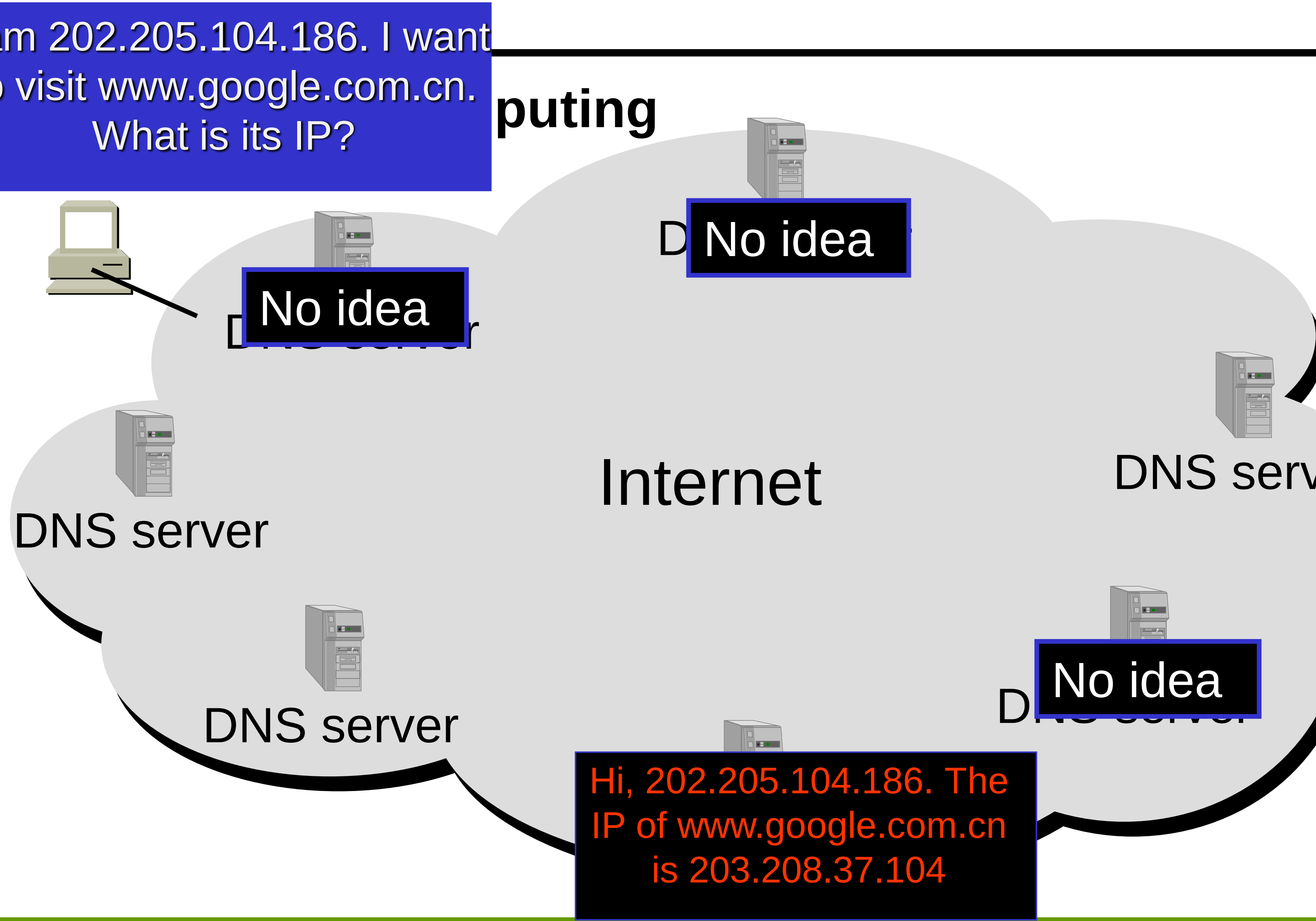


Figure 1: The high-level architecture of Borg. *Only a tiny fraction of the thousands of worker nodes are shown.*

□ 2 main architectur puting

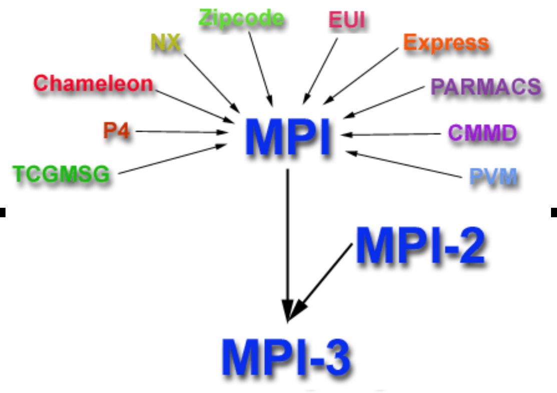
Peer-to-Peer



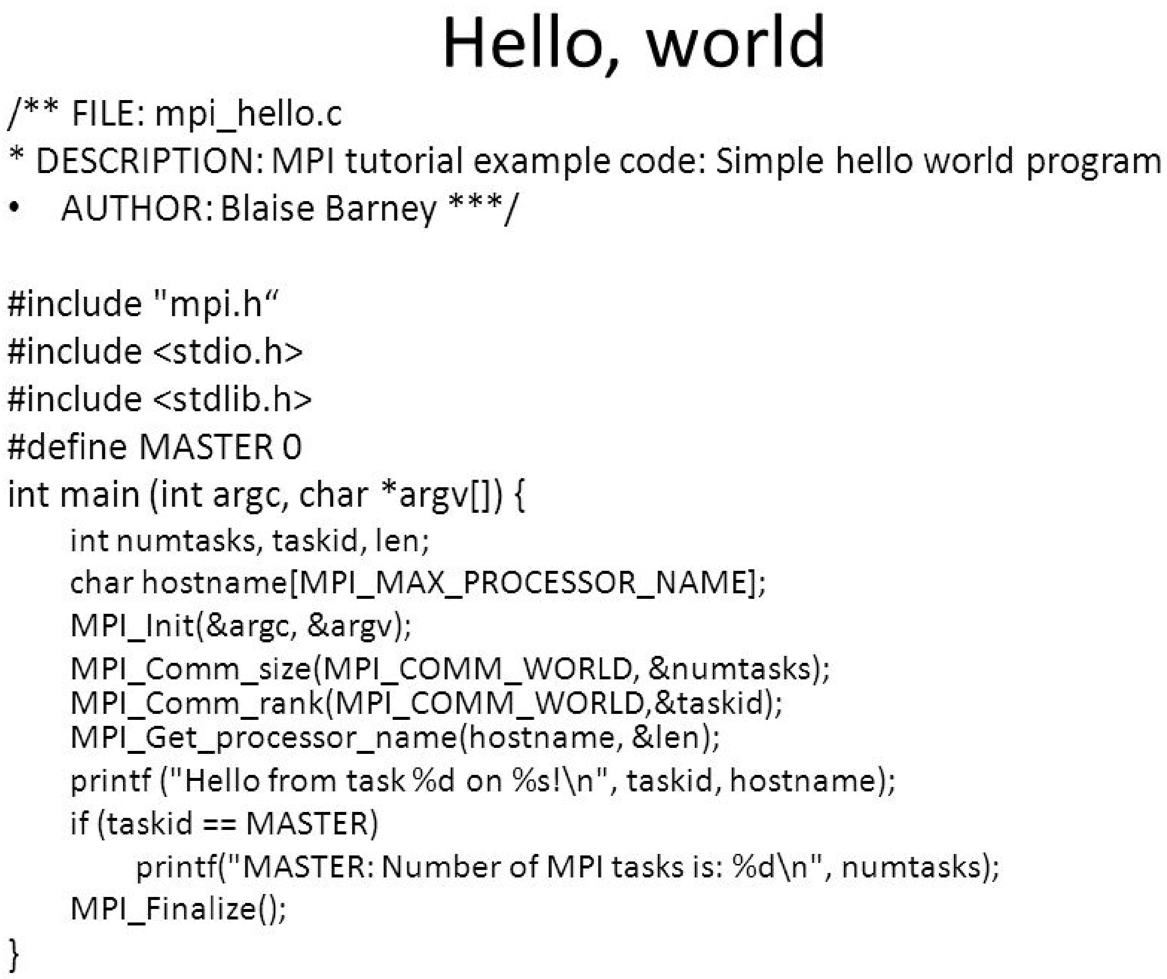

I am 202.205.104.186. I want to visit www.google.com.cn. What is its IP?

Hi, 202.205.104.186. The IP of www.google.com.cn is 203.208.37.104

HPC programming frameworks



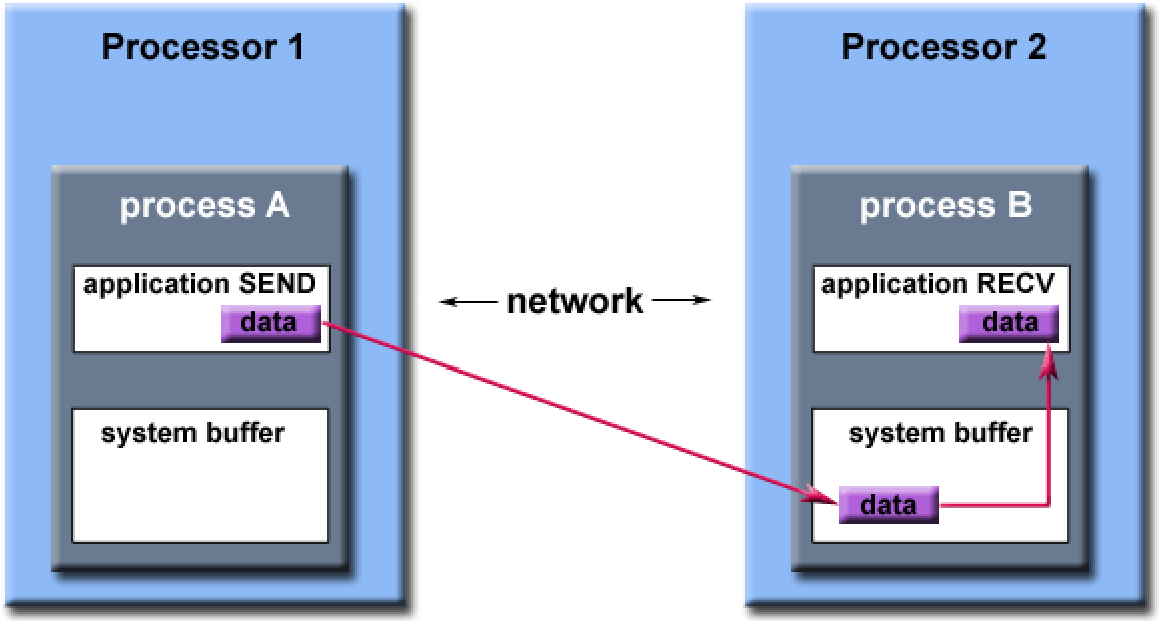
□ MPI – Message Passing Interface



Hello, world

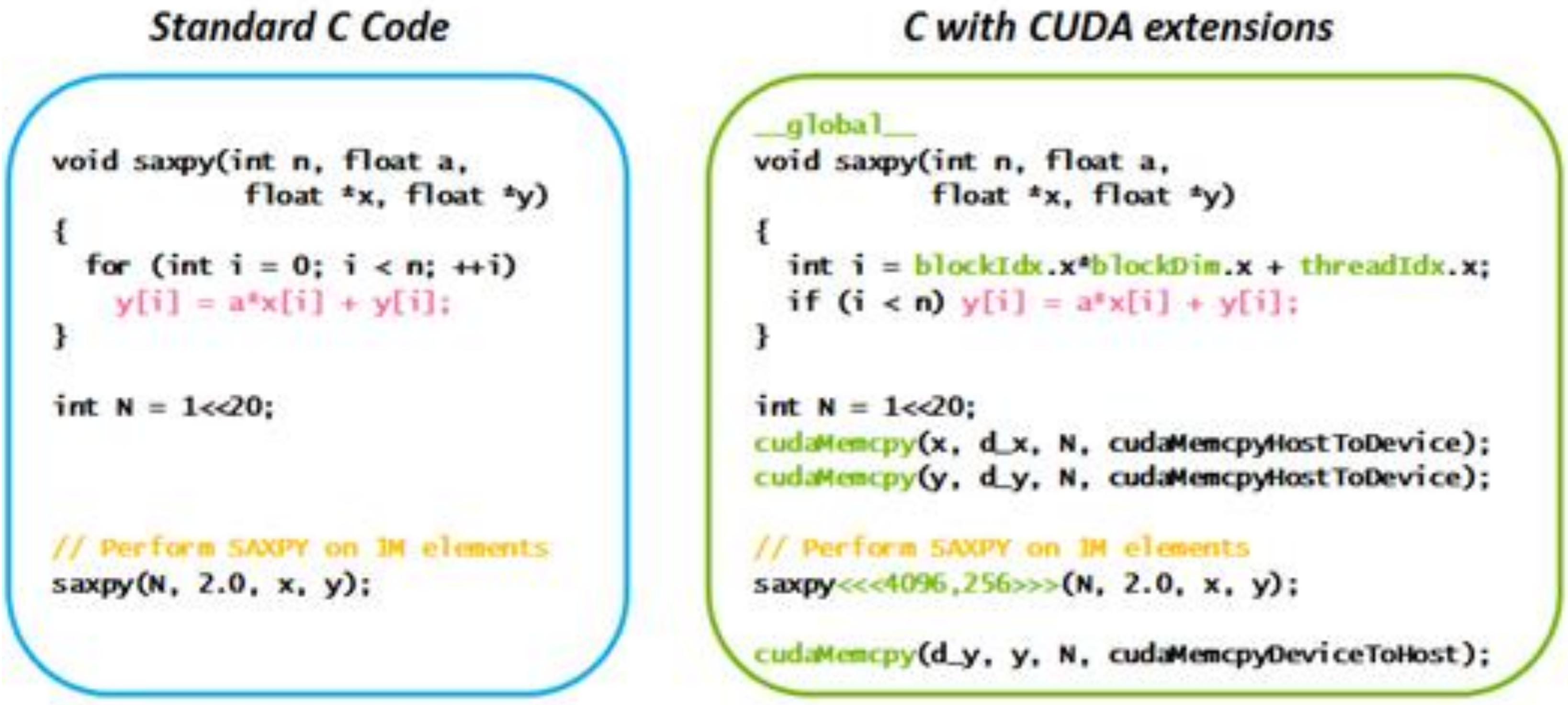
/** FILE: mpi_hello.c
* DESCRIPTION: MPI tutorial example code: Simple hello world program
- AUTHOR: Blaise Barney ***/

```
#include "mpi.h“
#include <stdio.h>
#include <stdlib.h>
#define MASTER 0
int main (int argc, char *argv[]) {
    int numtasks, taskid, len;
    char hostname[MPI_MAX_PROCESSOR_NAME];
    MPI_Init(&argc, &argv);
    MPI_Comm_size(MPI_COMM_WORLD, &numtasks);
    MPI_Comm_rank(MPI_COMM_WORLD,&taskid);
    MPI_Get_processor_name(hostname, &len);
    printf ("Hello from task %d on %s!\n", taskid, hostname);
    if (taskid == MASTER)
        printf("MASTER: Number of MPI tasks is: %d\n", numtasks);
    MPI_Finalize();
}
```



Path of a message buffered at the receiving process

□ GPU/CUDA – Graphic Processing Unit/ Compute Unified Device Architecture[统一计算架构]

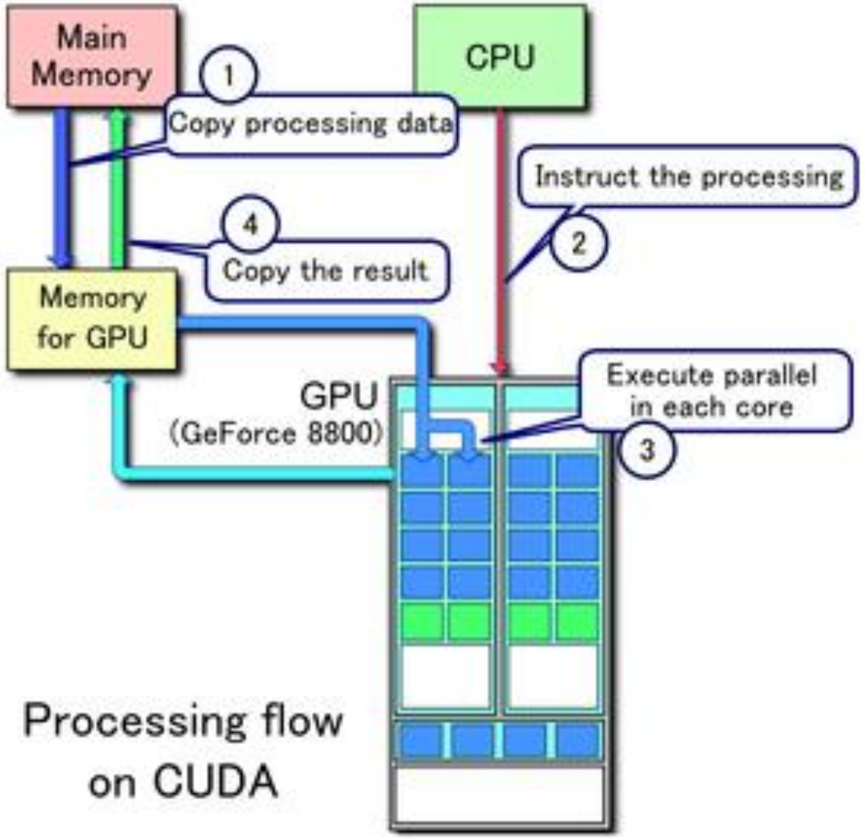


Standard C Code

```
void saxpy(int n, float a,
           float *x, float *y)
{
  for (int i = 0; i < n; ++i)
    y[i] = a*x[i] + y[i];
}

int N = 1<<20;



// Perform SAXPY on 1M elements
saxpy(N, 2.0, x, y);
```

C with CUDA extensions

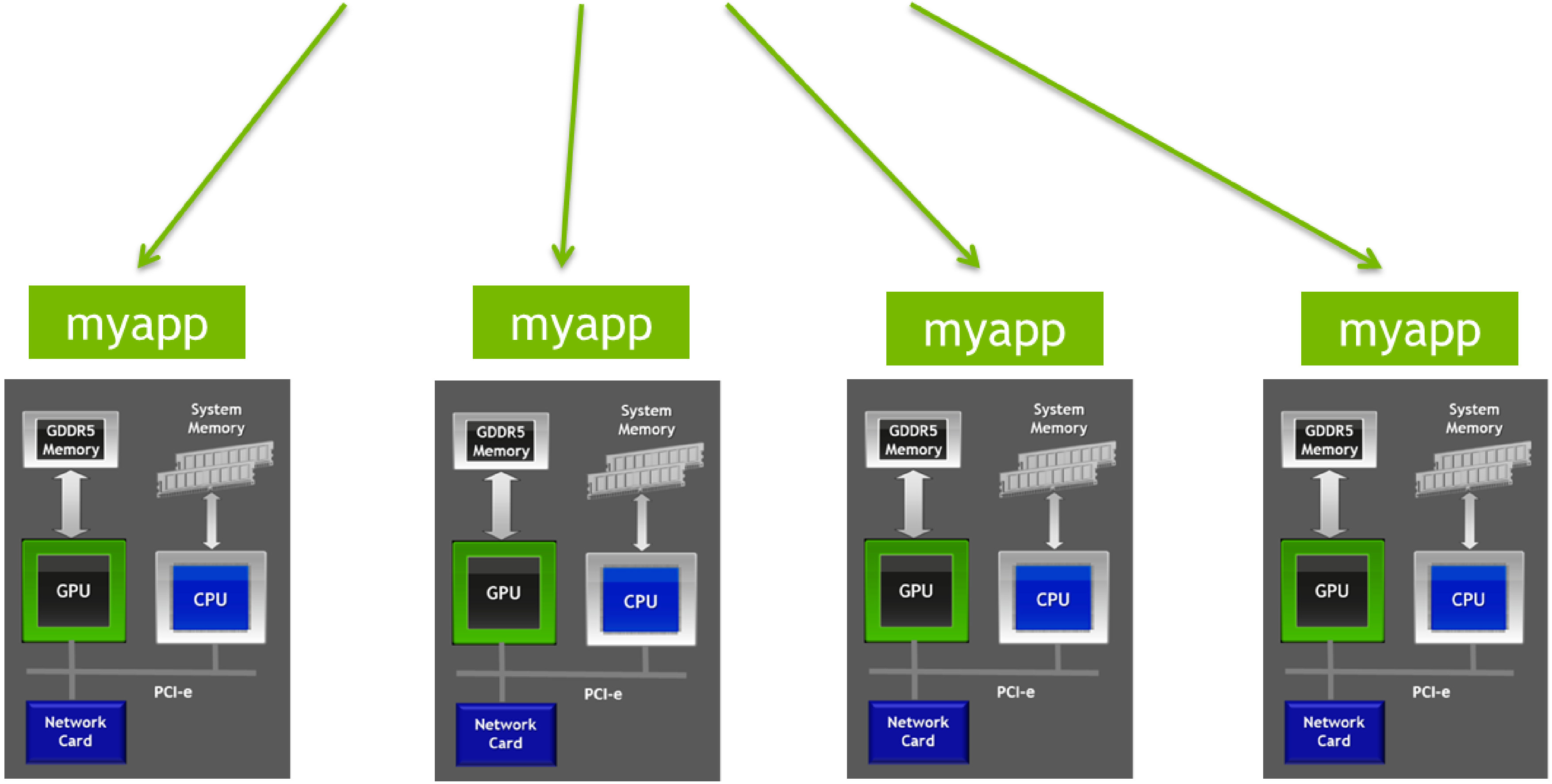
```
__global__
void saxpy(int n, float a,
           float *x, float *y)
{
  int i = blockIdx.x*blockDim.x + threadIdx.x;
  if (i < n) y[i] = a*x[i] + y[i];
}

int N = 1<<20;
cudaMemcpy(x, d_x, N, cudaMemcpyHostToDevice);
cudaMemcpy(y, d_y, N, cudaMemcpyHostToDevice);

// Perform SAXPY on 1M elements
saxpy<<<4096,256>>>(N, 2.0, x, y);

cudaMemcpy(d_y, y, N, cudaMemcpyDeviceToHost);
```



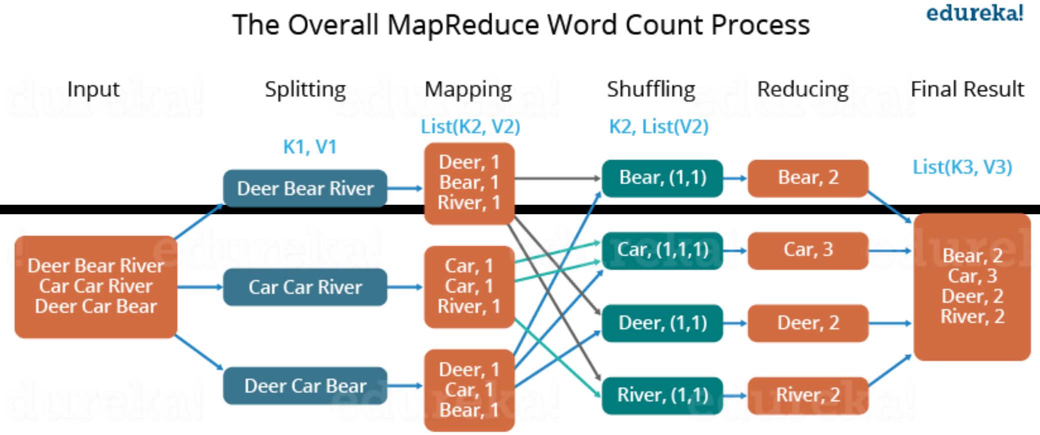
Processing flow on CUDA

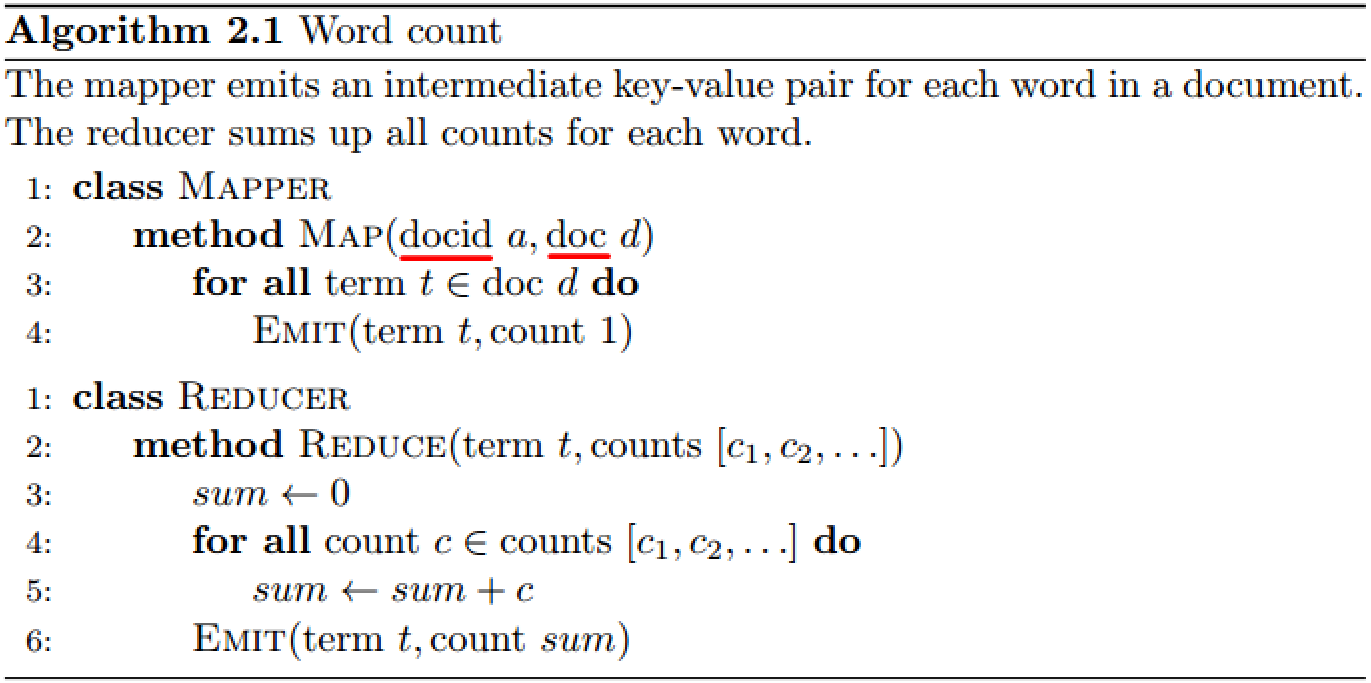
MPI on GPU

```
mpirun -np 4 ./myapp <args>
```



MR – Map/Reduce

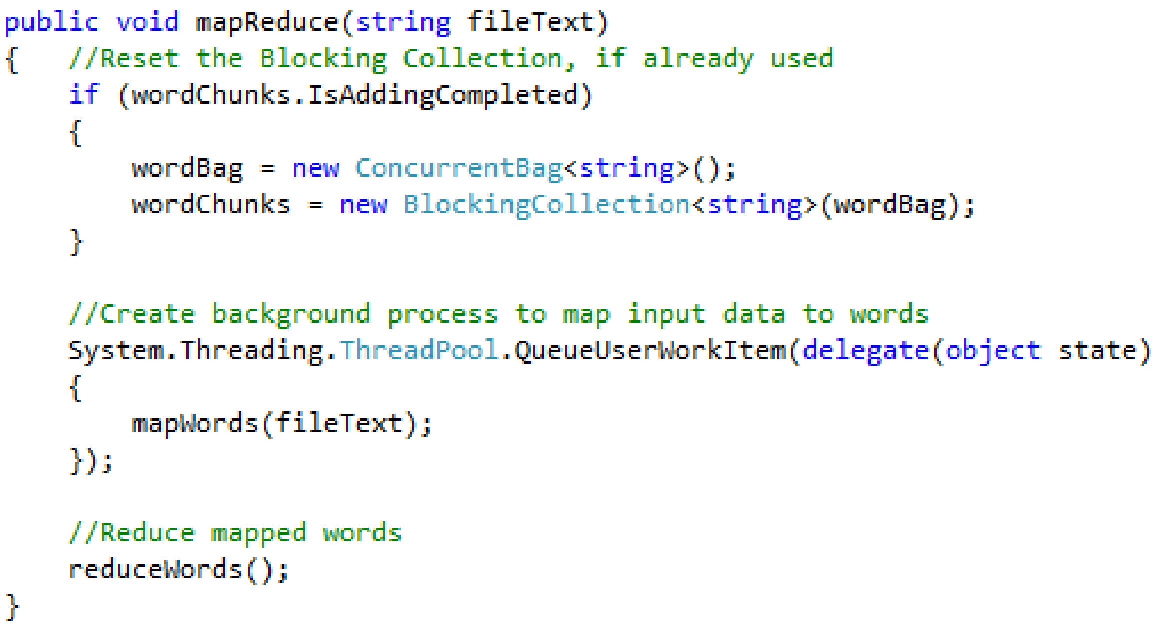




Algorithm 2.1 Word count

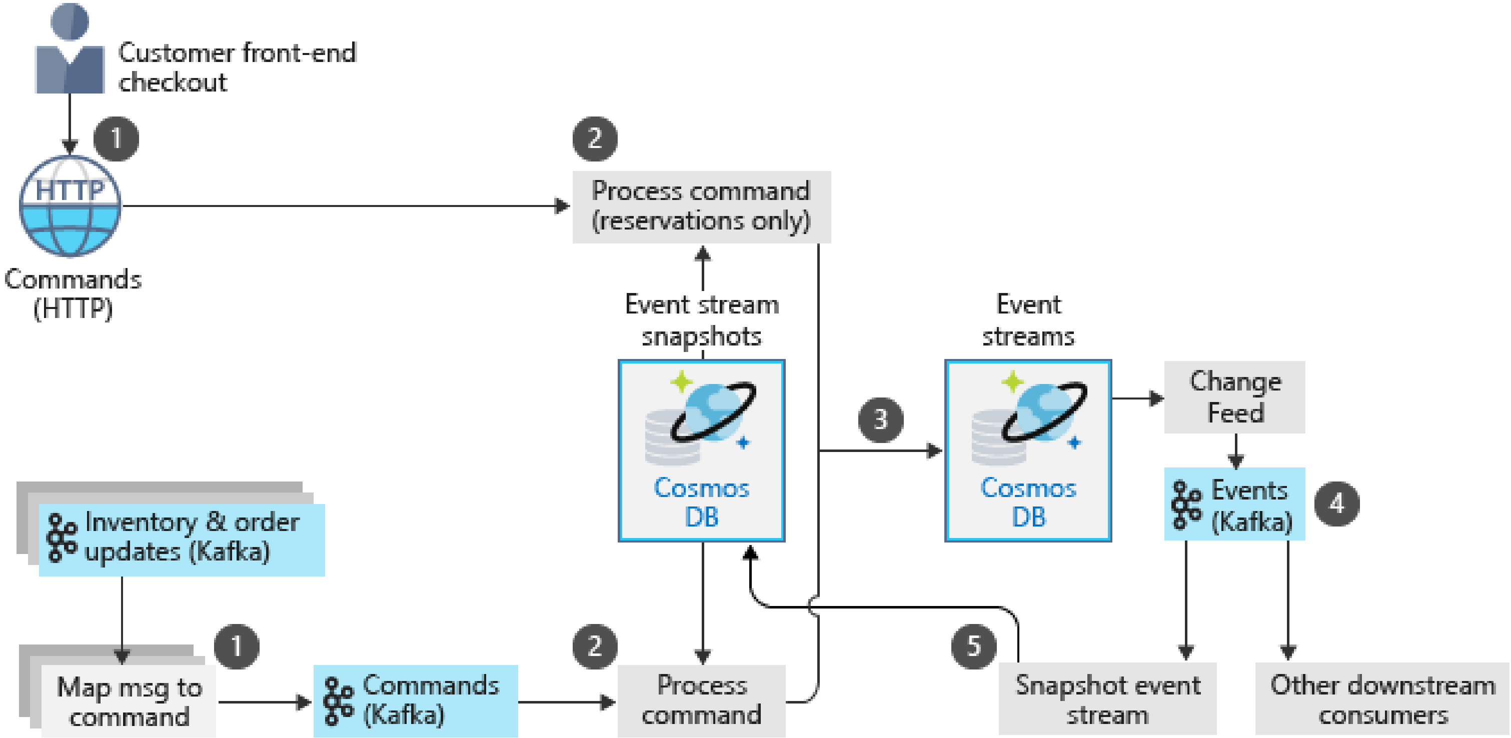
The mapper emits an intermediate key-value pair for each word in a document. The reducer sums up all counts for each word.

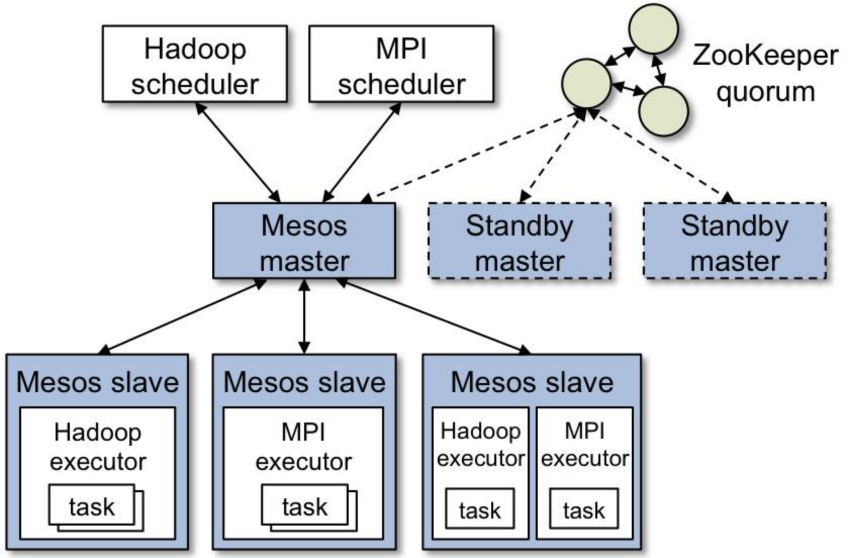
```
class Mapper
    method Map(docid $a$, doc $d$)
        for all term $t \in$ doc $d$ do
            Emit(term $t$, count 1)

class Reducer
    method Reduce(term $t$, counts $[c_1, c_2, \ldots]$)
        $sum \leftarrow 0$
        for all count $c \in$ counts $[c_1, c_2, \ldots]$ do
            $sum \leftarrow sum + c$
        Emit(term $t$, count $sum$)
```



```
public void mapReduce(string fileText)
{   //Reset the Blocking Collection, if already used
    if (wordChunks.IsAddingCompleted)
    {
        wordBag = new ConcurrentBag<string>();
        wordChunks = new BlockingCollection<string>(wordBag);
    }

    //Create background process to map input data to words
    System.Threading.ThreadPool.QueueUserWorkItem(delegate(object state)
    {
        mapWords(fileText);
    });

    //Reduce mapped words
    reduceWords();
}
```

Big Data





Mesos
2009 (while still named Nexus)

Many Open Source programs proposed now by CHINA

- 2012 HuaWei
  (from **PostgreSQL** 9.2)
  (行、列存储;多线程)
  (OLTP, OLAP, HTAP)
  (MPP, Cluster/Distributed)
  (Native AI DB)

- 2011 Pivotal
  HAWQ (Big Data)
  (forked from Greenplum 4.2.0)
  (外链接方式可访问HDFS,
  从而支持了Big Data – 可
  访问HDFS)
  2017 Apache 顶级

- Kylin **Bring OLAP Back to Big Data!**
  2015年Apache 顶级
  http://kylin.apache.org/

- 2016 ShardingSphere (当当和京东)
  一款分布式数据库中间件
  (2020年4月16日成为 Apache顶级项目)

- 2019 HuDi (Hadoop Updates and Incrementals)
  由Uber开发并开源的Data Lakes解决方案, 能够
  摄入（Ingest）和管理（Manage）基于HDFS之
  上的大型分析数据集，主要目的是高效的减少入
  库延时

- 2017 TiKV distributed transactional
  key-value database
  * TiDB(based on TiKV)
  分布式数据库文件管理器,支持混
  合事务和分析处理(HTAP)

- 2020 Ozone 腾讯云主导的分布式对象存储系统
  可用于小文件和大文件存储
  2020年成为Apache 顶级项目

In short

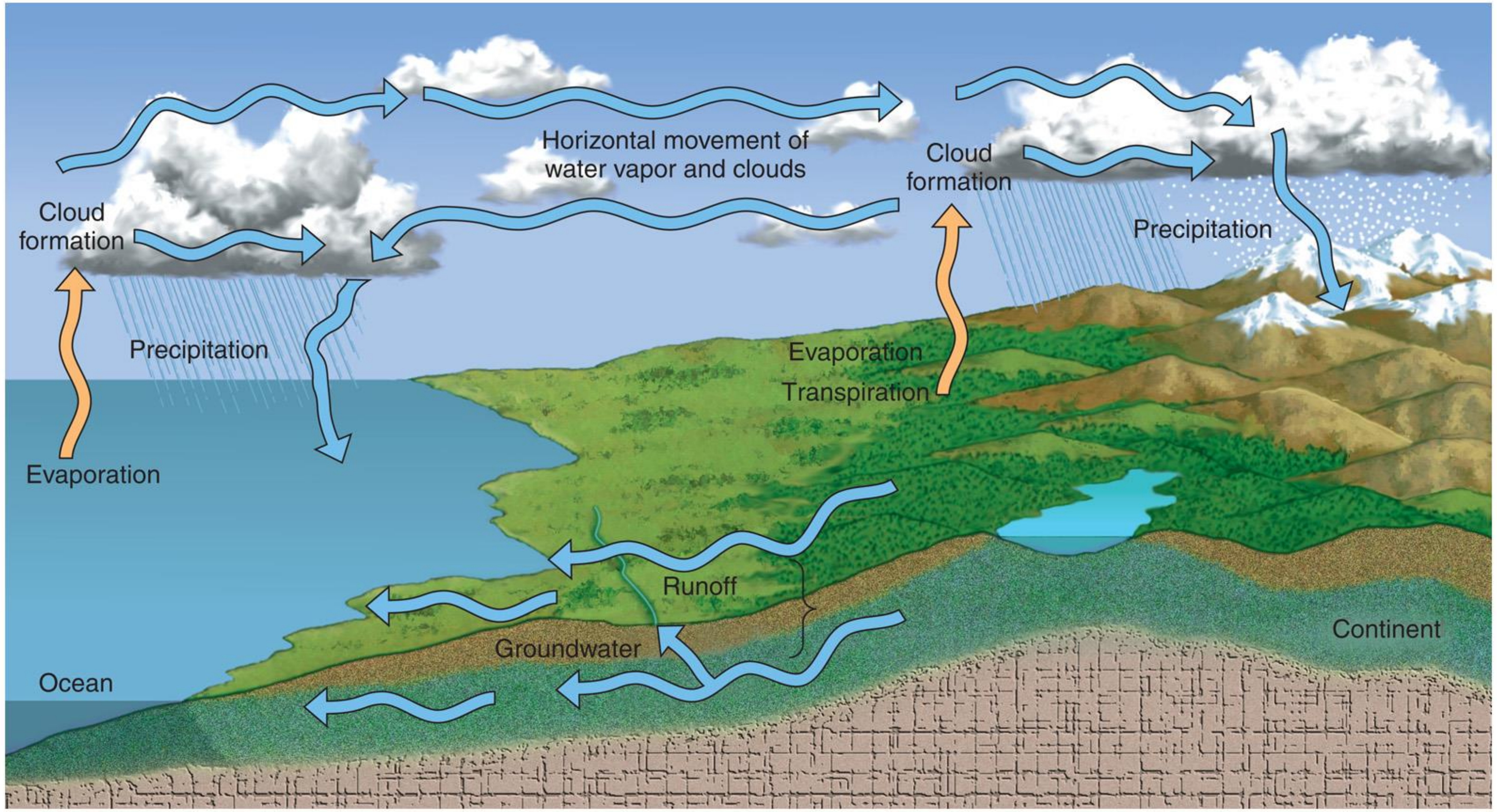
1. **You have to convert your program into EUs**
   - **DAOM**/PCAM
2. **Choose environment to finish EUs**
   - Systems
     - **P**arallel: multi-processor system – Multi-Core, GPU, MPP, …
     - **D**istributed: Cluster
   - Frameworks
     - Data parallel, SAS (Shared Address Space), Message passing
     - **MPI** (P or D), **CUDA** (P), **MR** (D), **Spark**, **Graph computing**, …
3. **Execute**

大商务，需要大数据

- **In IT age, <u>platform sticking consumers</u> [黏着客户的平台] is the popular pattern for e-business – Amazon, Google, Alibaba, JD, …**
  - **Large scale** (data and computing power) is important, which needs HPC
- **Large Scale Data – Big Data**
  - From File to Big Data
- **Large Scale computing power – High Performance Computing is now popular for business**
  - According to Top 500, MPP and Cluster
- **Additional bonus for Scientific Computing**
  - Weather forecasting

Weather forecasting is the 1st problem requiring HPC

- The Hydrologic Cycle



© 2010 Pearson Education, Inc.

Vilhelm Bjerknes’ Vision

- **1901** – Wanted to incorporate physics into weather forecasting
  - Start with complete set of initial conditions (3-D)
  - Solve equations using graphical methods
  - Initial state not sufficient for good forecasts
  - Did not use continuity equation to derive the initial vertical wind component (no direct measurements available)

Source: Historical Essays on Meteorology 1919-1995, AMS

PDE is the math model for WF – Bu， No calculus solution!

- **Atmosphere dynamics with considering many parameters:**
  - 3-D: Temperature, Humidity, Wind speed and Wind direction, Atmospheric Pressure,
  - Later even Dew Point, Relative Humidity …

$$\begin{cases}
\dfrac{du}{dt}-\dfrac{uv\tan\varphi}{r}+\dfrac{uw}{r}=-\dfrac{1}{\rho}\dfrac{\partial p}{r\cos\varphi\partial\lambda}+fv-\tilde{f}w+F_\lambda \\
\dfrac{dv}{dt}+\dfrac{u^2\tan\varphi}{r}+\dfrac{vw}{r}=-\dfrac{1}{\rho}\dfrac{\partial p}{r\partial\varphi}-fu+F_\varphi \\
\dfrac{dw}{dt}-\dfrac{u^2+v^2}{r}=-\dfrac{1}{\rho}\dfrac{\partial p}{\partial r}-g+\tilde{f}u+F_r \\
\dfrac{d\rho}{dt}+\rho\left[\dfrac{1}{r\cos\varphi}\dfrac{\partial u}{\partial\lambda}+\dfrac{1}{r\cos\varphi}\dfrac{\partial(v\cos\varphi)}{\partial\varphi}+\dfrac{\partial(wr^2)}{r^2\partial r}\right]=0 \\
\dfrac{dT}{dt}-\dfrac{RT}{C_p p}\dfrac{dp}{dt}=\dfrac{\dot{Q}}{C_p} \\
p=\rho RT
\end{cases}$$

Numeric method

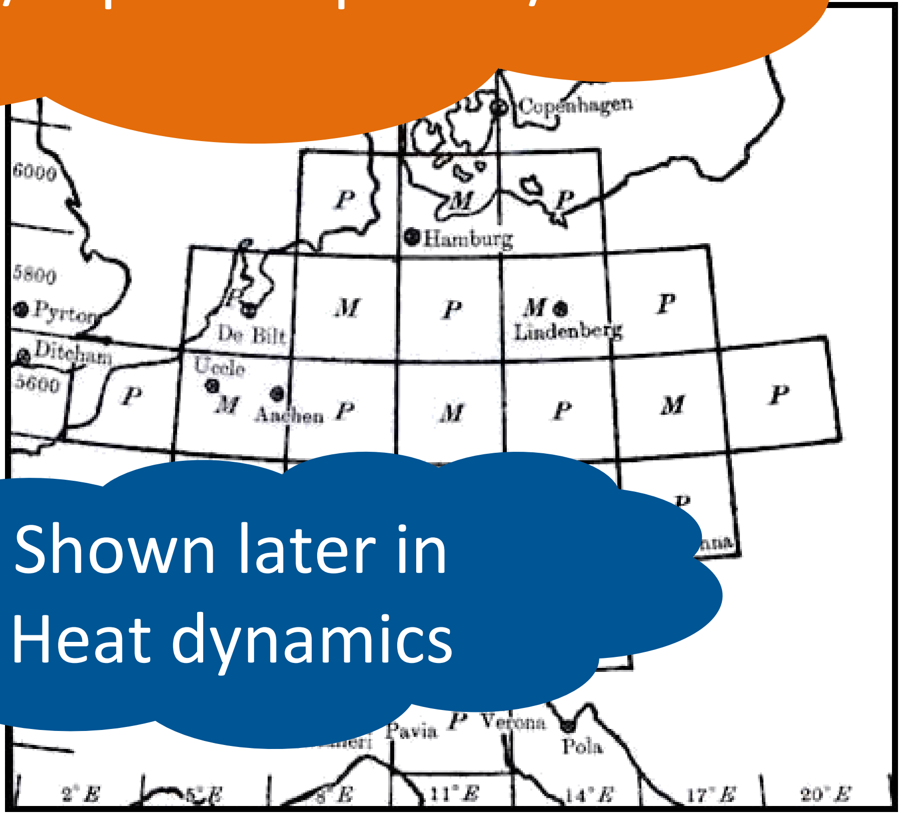
Yes, we need numeric weather prediction – by using computers/supercomputers/HPC

First **numerical forecast** made in 192
Lewis Fry Richardson.

Took several months, calculating by hand, to produce a 6-hour forecast.

It failed…badly!

But, it demonstrated the means of producing quantitative forecasts. Its failure has since been shown to be due to the limited understanding of some atmospheric processes at the time.

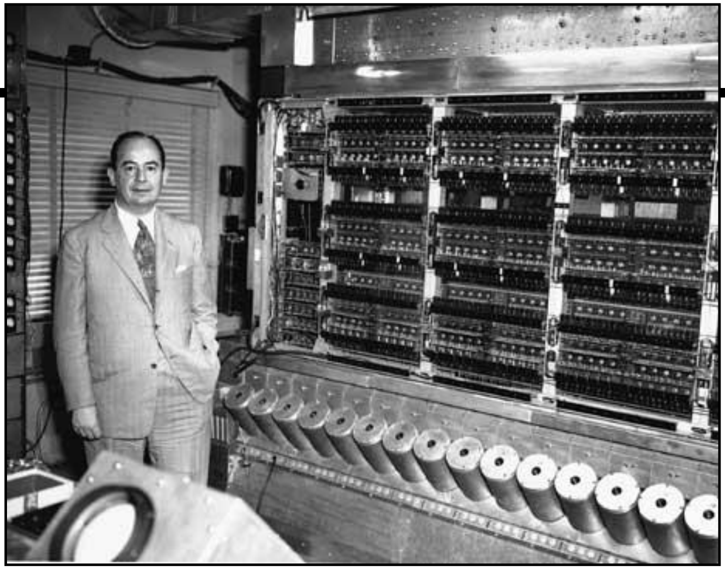


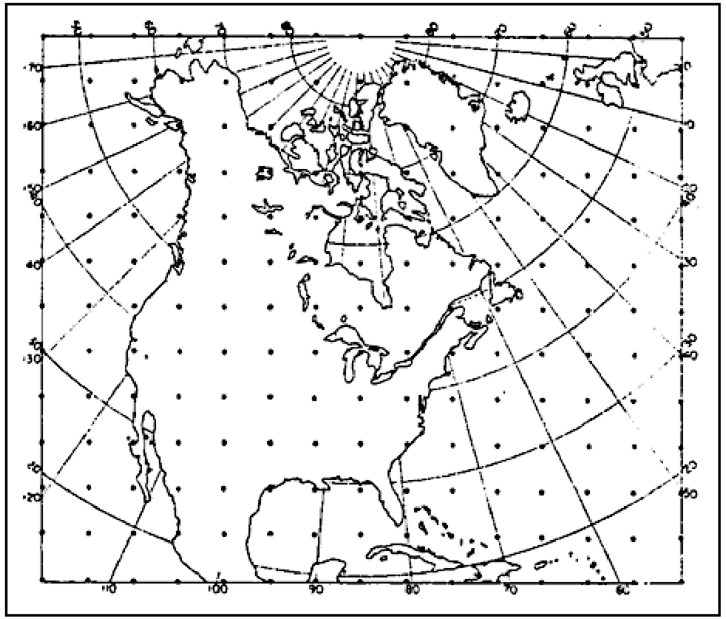
Shown later in Heat dynamics

L. F. Richardson's computational grid: Pressure is determined in squares marked 'P', momentum in those marked 'M'.

First successful forecast: 1950 by Jule Charney, Fjörtoft, and von Neumann, using ENIAC.

A 24-hour forecast took 33 days to produce, working day and night.

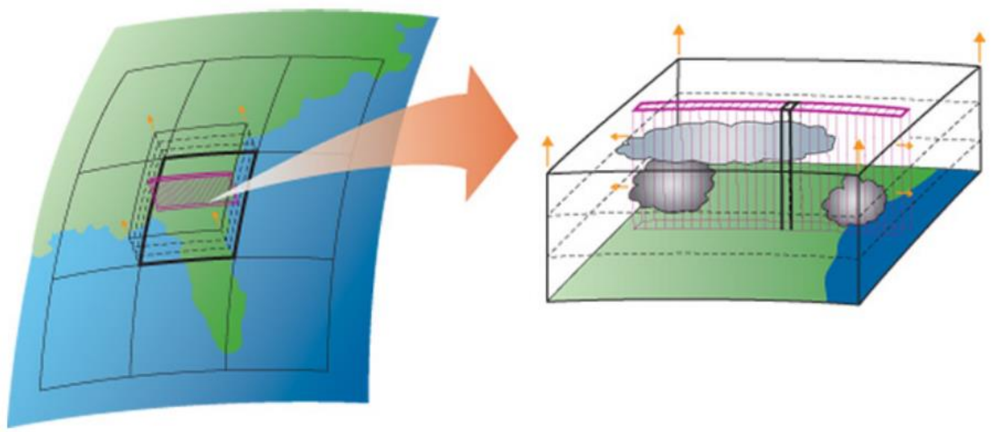




https://en.wikipedia.org/wiki/Numerical_weather_prediction

The way to carry out numeric method

- **A grid is used to divide the region of interest.**
  - Since the PDE is satisfied at each point in the area, it must be satisfied at each point of the grid.



- **A finite difference approximation is obtained at each grid point.**

$$\frac{\partial^2\ T(x,y)}{\partial x^2} \approx \frac{T_{i+1,j} - 2T_{i,j} + T_{i-1,j}}{(\Delta x)^2}, \quad \frac{\partial^2\ T(x,y)}{\partial y^2} \approx \frac{T_{i,j+1} - 2T_{i,j} + T_{i,j-1}}{(\Delta y)^2}$$

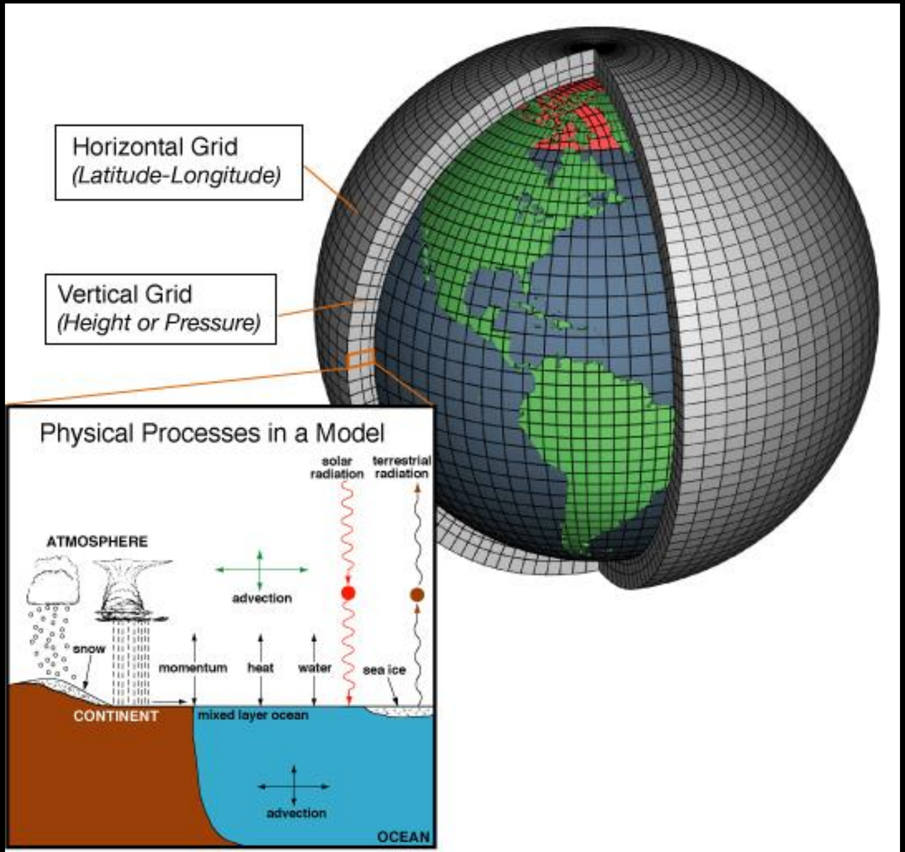
$$\frac{\partial^2 T(x,y)}{\partial x^2} = \frac{T_{i+1,j} - 2T_{i,j} + T_{i-1,j}}{(\Delta x)^2},$$

$$\frac{\partial^2 T(x,y)}{\partial y^2} = \frac{T_{i,j+1} - 2T_{i,j} + T_{i,j-1}}{(\Delta y)^2}$$

$$\Rightarrow \frac{\partial^2 T(x,y)}{\partial x^2} + \frac{\partial^2 T(x,y)}{\partial y^2} = 0$$

is approximated by:

$$\frac{T_{i+1,j} - 2T_{i,j} + T_{i-1,j}}{(\Delta x)^2} + \frac{T_{i,j+1} - 2T_{i,j} + T_{i,j-1}}{(\Delta y)^2} = 0$$



$$\frac{T_{i+1,j} - 2T_{i,j} + T_{i-1,j}}{(\Delta x)^2} + \frac{T_{i,j+1} - 2T_{i,j} + T_{i,j-1}}{(\Delta y)^2} = 0$$

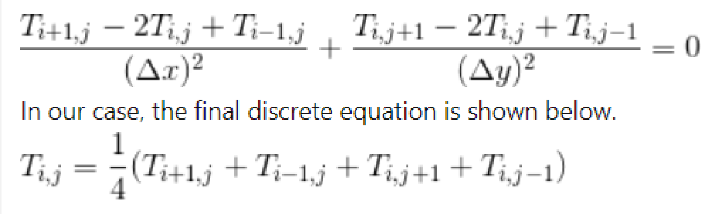
$(Laplacian\ Difference\ Equation)$

$Assume: \Delta x = \Delta y = h$

$$\Rightarrow T_{i+1,j} + T_{i-1,j} + T_{i,j+1} + T_{i,j-1} - 4T_{i,j} = 0$$

The data is so HUGE which cannot be processed with one Computer!

Global Grid (0.25*0.25 → 1440X720 dots = 62208000 = 6.22*10^7)



$$\frac{T_{i+1,j}-2T_{i,j}+T_{i-1,j}}{(\Delta x)^2}+\frac{T_{i,j+1}-2T_{i,j}+T_{i,j-1}}{(\Delta y)^2}=0$$

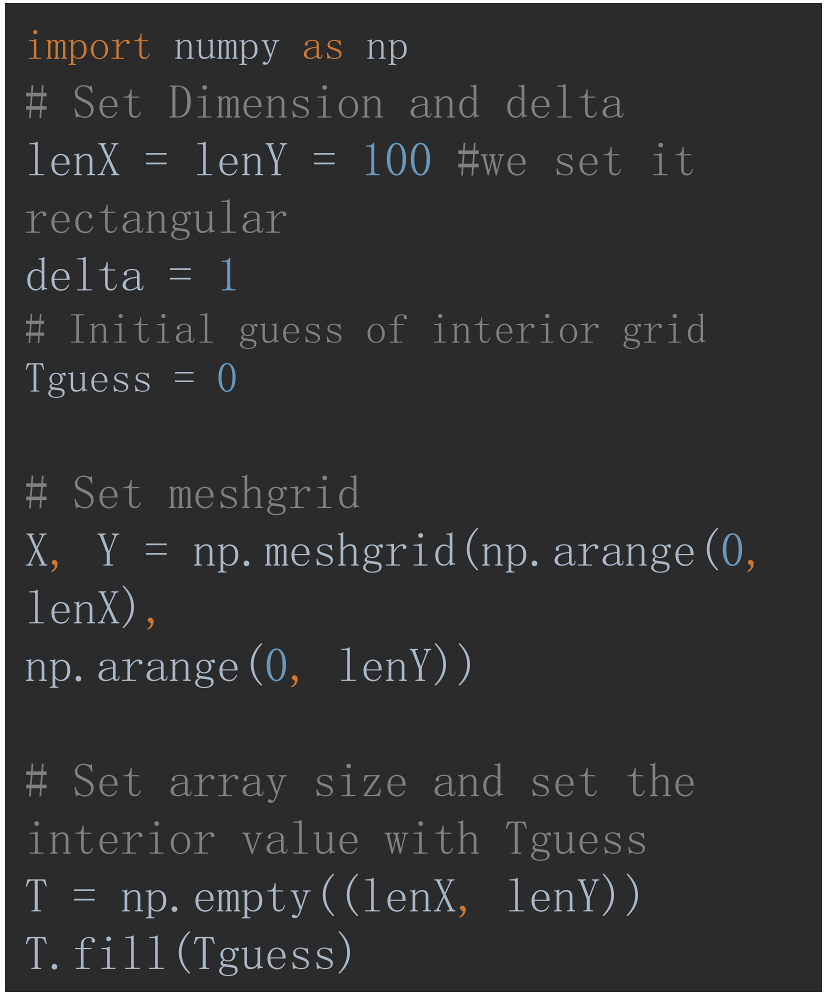
In our case, the final discrete equation is shown below.

$$T_{i,j}=\frac{1}{4}(T_{i+1,j}+T_{i-1,j}+T_{i,j+1}+T_{i,j-1})$$

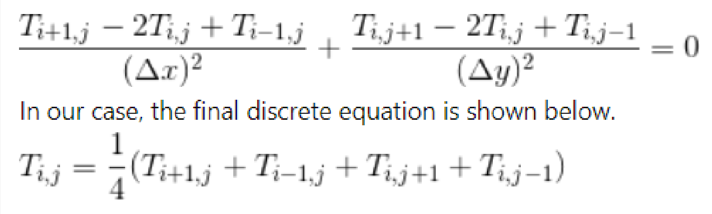
□ The code demonstration of “Using Python to Solve Computational Physics Problems”

1. Configure the parameters

- GRID
  - With Initial values [初始值]
  - Boundary conditions
    - ✓ [边界条件]
- Termination condition
  - Iteration number or Epsilon



```python
import numpy as np
# Set Dimension and delta
lenX = lenY = 100 #we set it rectangular
delta = 1
# Initial guess of interior grid
Tguess = 0

# Set meshgrid
X, Y = np.meshgrid(np.arange(0, lenX), np.arange(0, lenY))

# Set array size and set the interior value with Tguess
T = np.empty((lenX, lenY))
T.fill(Tguess)
```



$$\frac{T_{i+1,j} - 2T_{i,j} + T_{i-1,j}}{(\Delta x)^2} + \frac{T_{i,j+1} - 2T_{i,j} + T_{i,j-1}}{(\Delta y)^2} = 0$$

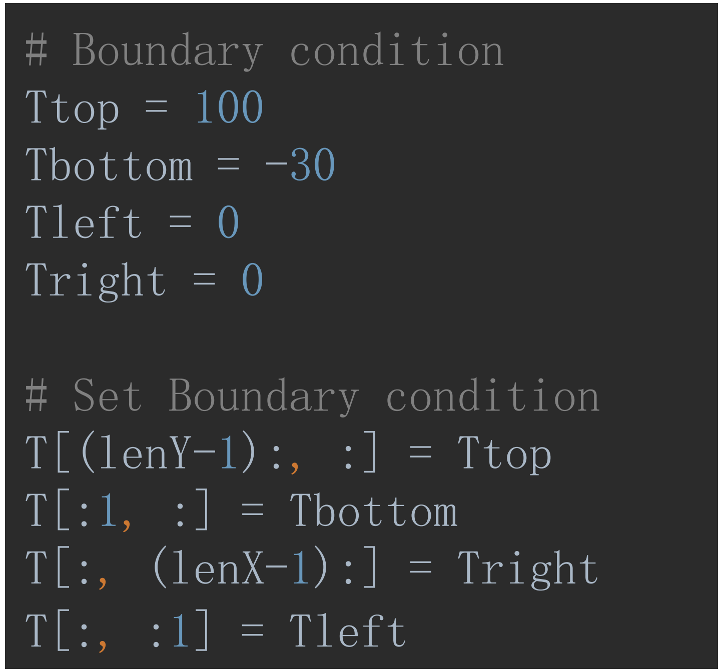
In our case, the final discrete equation is shown below.

$$T_{i,j} = \frac{1}{4}(T_{i+1,j} + T_{i-1,j} + T_{i,j+1} + T_{i,j-1})$$

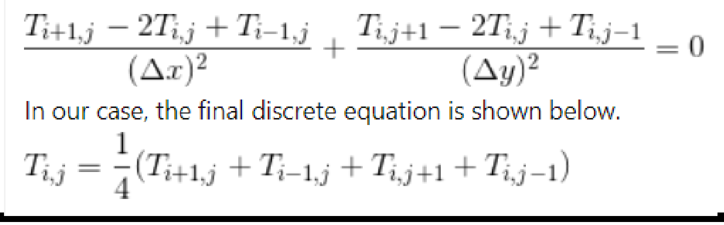
□ The code demonstration of "Using Python to Solve Computational Physics Problems"

1. Configure the parameters

- GRID
  - With Initial values [初始值]
  - With Boundary conditions
    - ✓ [边界条件]
- Termination condition
  - Iteration number or Epsilon



```
# Boundary condition
Ttop = 100
Tbottom = -30
Tleft = 0
Tright = 0

# Set Boundary condition
T[(lenY-1):, :] = Ttop
T[:1, :] = Tbottom
T[:, (lenX-1):] = Tright
T[:, :1] = Tleft
```



$$\frac{T_{i+1,j}-2T_{i,j}+T_{i-1,j}}{(\Delta x)^2}+\frac{T_{i,j+1}-2T_{i,j}+T_{i,j-1}}{(\Delta y)^2}=0$$

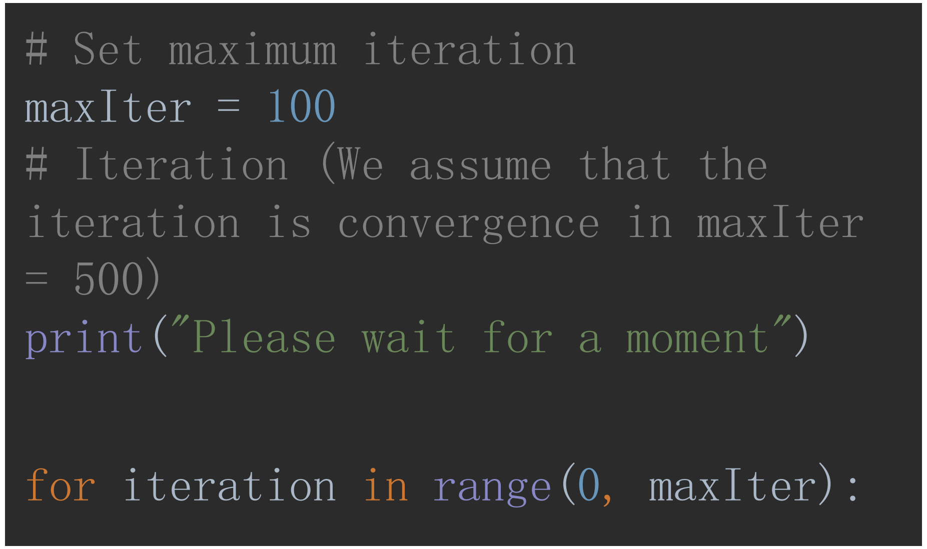
In our case, the final discrete equation is shown below.

$$T_{i,j}=\frac{1}{4}(T_{i+1,j}+T_{i-1,j}+T_{i,j+1}+T_{i,j-1})$$

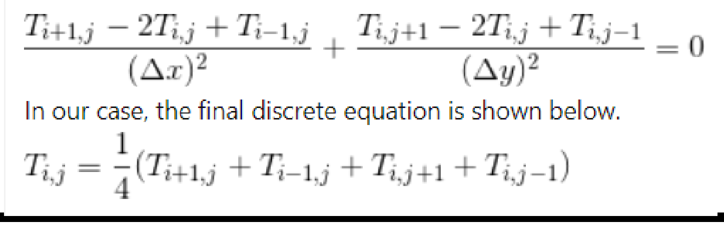
□ The code demonstration of “Using Python to Solve Computational Physics Problems”

1. Configure the parameters

- GRID
  - ➢With Initial values [初始值]
  - ➢With Boundary conditions
    - ✓[边界条件]
- Termination condition
  - ➢Iteration number or Epsilon



```
# Set maximum iteration
maxIter = 100
# Iteration (We assume that the iteration is convergence in maxIter = 500)
print("Please wait for a moment")


for iteration in range(0, maxIter):
```



$$\frac{T_{i+1,j}-2T_{i,j}+T_{i-1,j}}{(\Delta x)^2}+\frac{T_{i,j+1}-2T_{i,j}+T_{i,j-1}}{(\Delta y)^2}=0$$

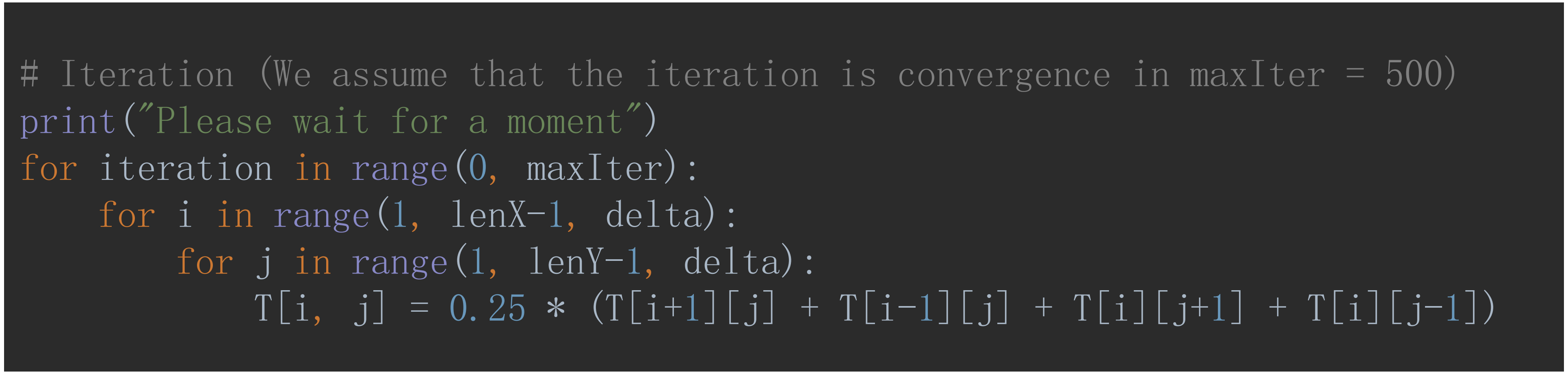
In our case, the final discrete equation is shown below.

$$T_{i,j}=\frac{1}{4}(T_{i+1,j}+T_{i-1,j}+T_{i,j+1}+T_{i,j-1})$$

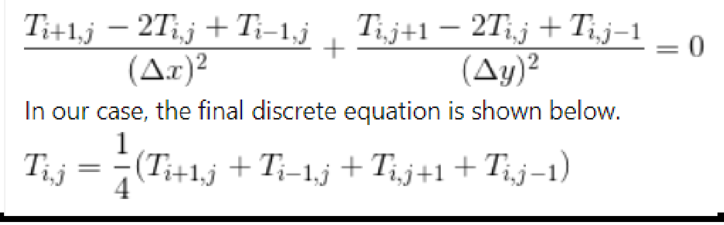
The code demonstration of “Using Python to Solve Computational Physics Problems”

2. Iterative updating

- Use “Termination condition” to control the updating of the internal vertices



```
# Iteration (We assume that the iteration is convergence in maxIter = 500)
print(″Please wait for a moment″)
for iteration in range(0, maxIter):
    for i in range(1, lenX-1, delta):
        for j in range(1, lenY-1, delta):
            T[i, j] = 0.25 * (T[i+1][j] + T[i-1][j] + T[i][j+1] + T[i][j-1])
```



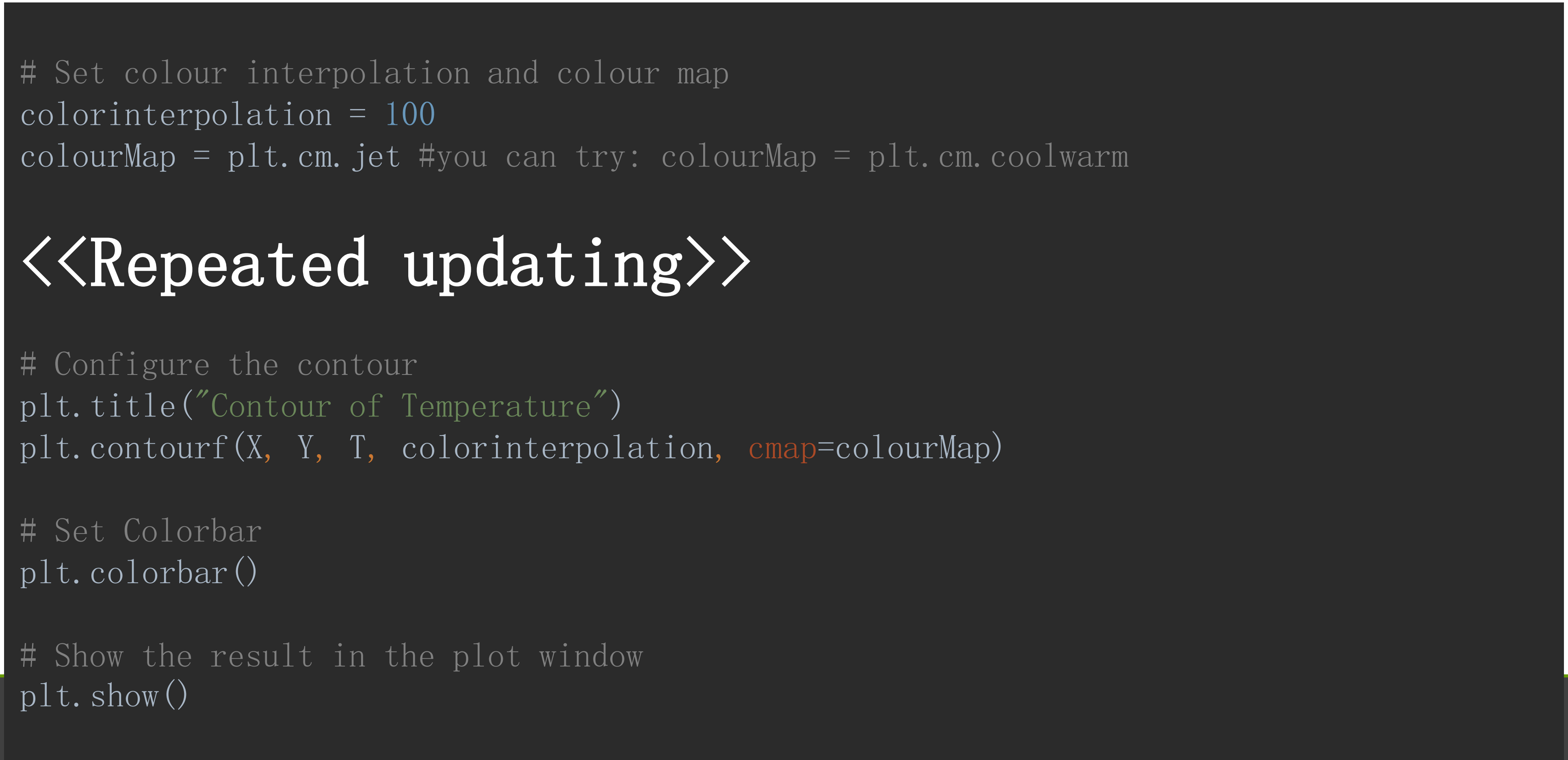
$$\frac{T_{i+1,j}-2T_{i,j}+T_{i-1,j}}{(\Delta x)^2}+\frac{T_{i,j+1}-2T_{i,j}+T_{i,j-1}}{(\Delta y)^2}=0$$

In our case, the final discrete equation is shown below.

$$T_{i,j}=\frac{1}{4}(T_{i+1,j}+T_{i-1,j}+T_{i,j+1}+T_{i,j-1})$$

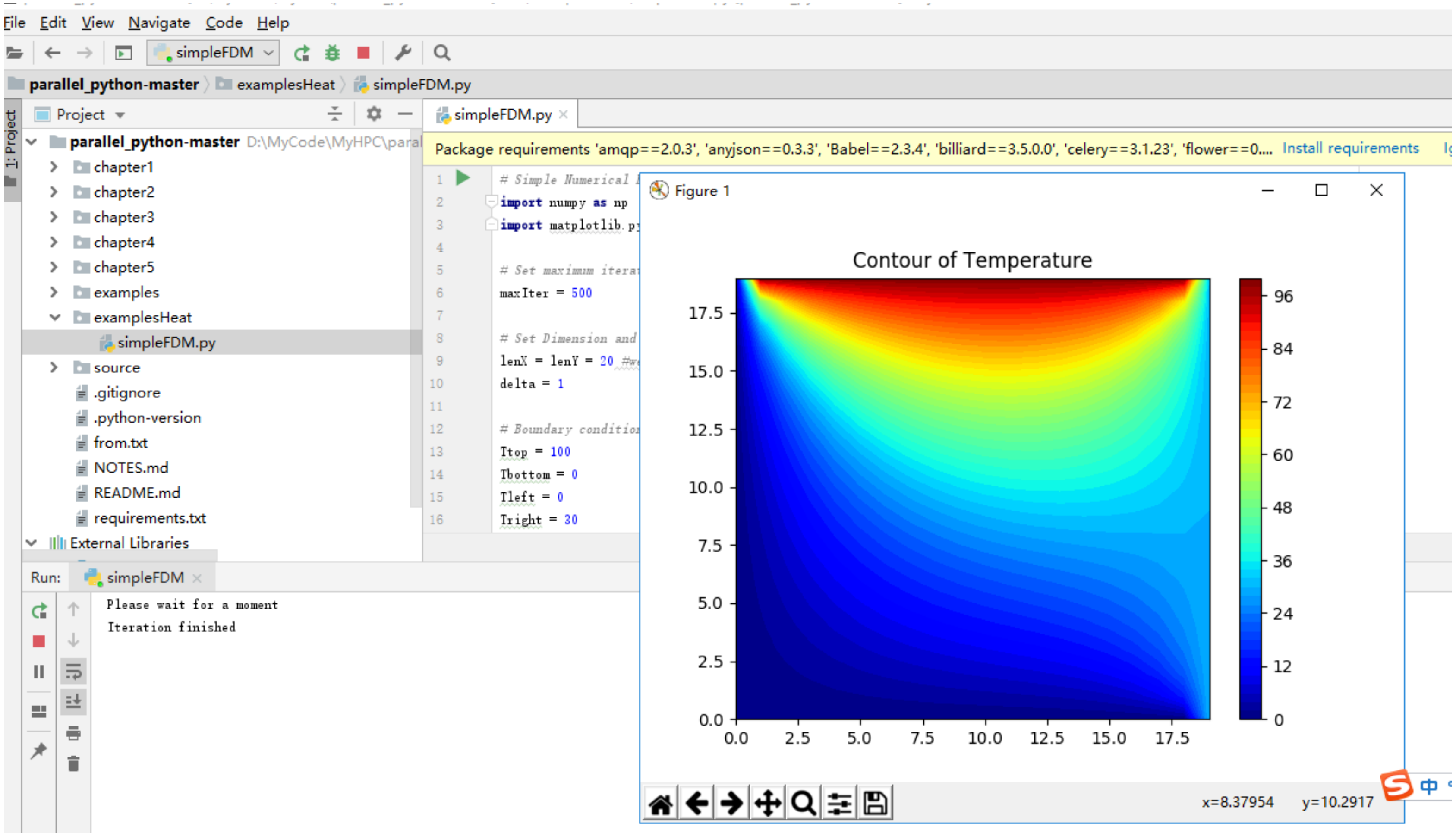
□ The code demonstration of “Using Python to Solve Computational Physics Problems”

3. Visualize the dynamics



```
# Set colour interpolation and colour map
colorinterpolation = 100
colourMap = plt.cm.jet #you can try: colourMap = plt.cm.coolwarm

<<Repeated updating>>

# Configure the contour
plt.title(”Contour of Temperature”)
plt.contourf(X, Y, T, colorinterpolation, cmap=colourMap)

# Set Colorbar
plt.colorbar()

# Show the result in the plot window
plt.show()
```

□ Copy the code into PyCharm project

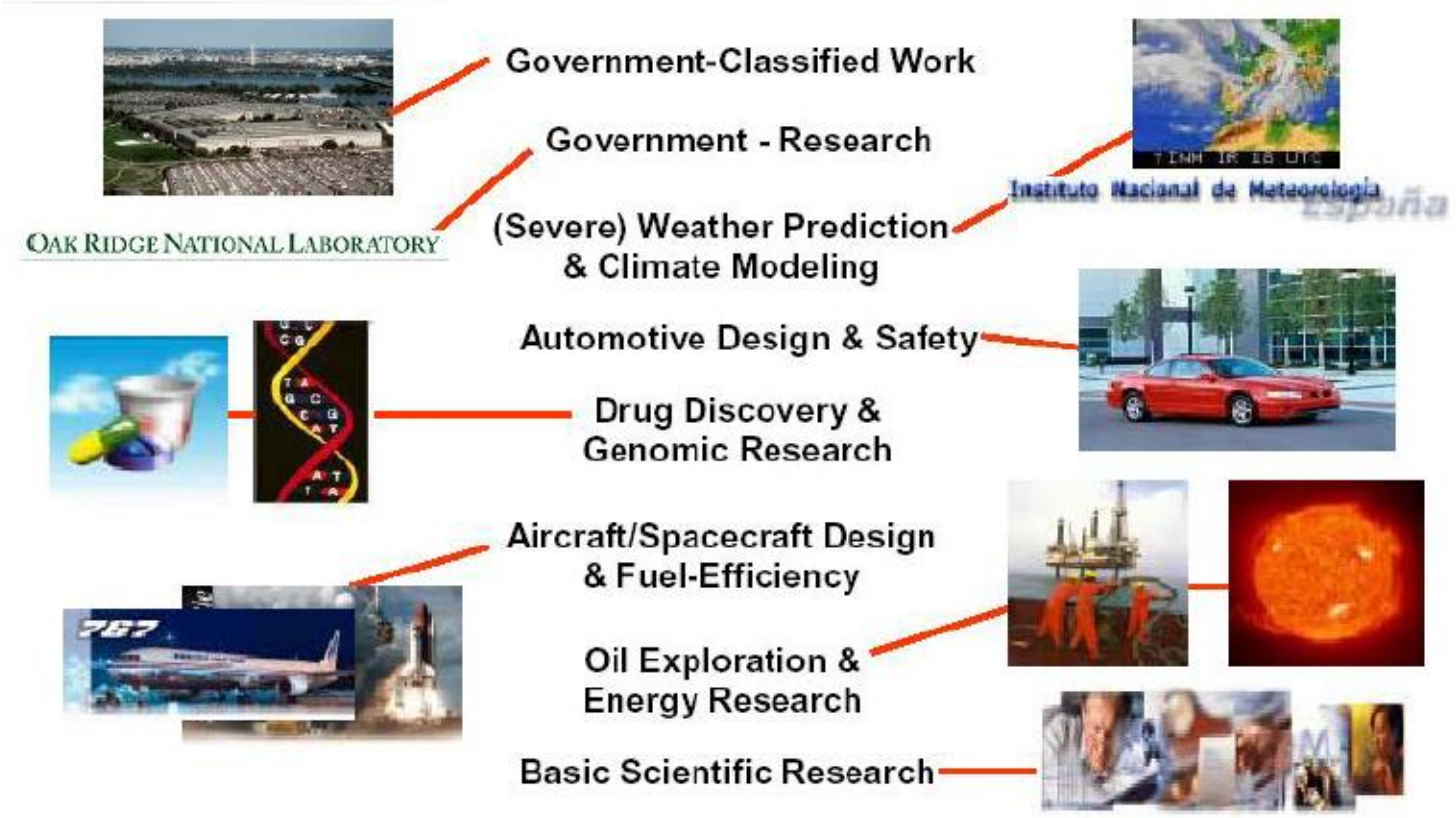


Package requirements 'amqp==2.0.3', 'anyjson==0.3.3', 'Babel==2.3.4', 'billiard==3.5.0.0', 'celery==3.1.23', 'flower==0....

```
# Simple Numerical I
import numpy as np
import matplotlib.py

# Set maximum itera
maxIter = 500

# Set Dimension and
lenX = lenY = 20 #w
delta = 1

# Boundary conditio
Ttop = 100
Tbottom = 0
Tleft = 0
Tright = 30
```

```
Please wait for a moment
Iteration finished
```

BTW, there are still many other scientific problems!


OAK RIDGE NATIONAL LABORATORY
INM IR 18 UTC
Instituto Nacional de Meteorologia
España
767

Government-Classified Work

Government - Research

(Severe) Weather Prediction & Climate Modeling

Automotive Design & Safety

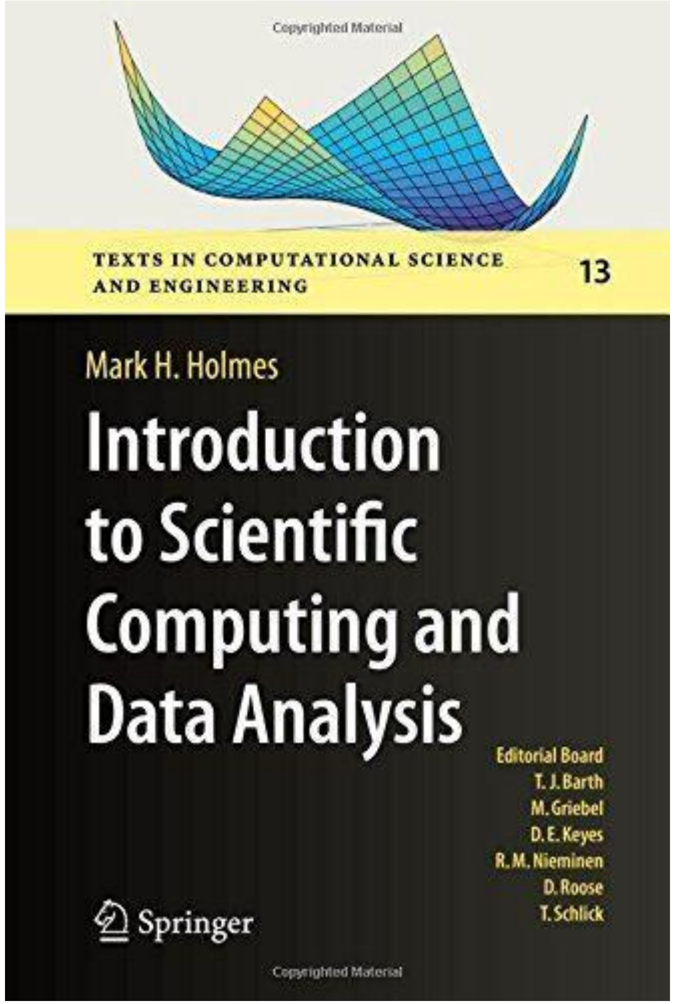
Drug Discovery & Genomic Research

Aircraft/Spacecraft Design & Fuel-Efficiency

Oil Exploration & Energy Research

Basic Scientific Research

- 偏微分方程数值解法
- 第二版
- 陆金甫，关治



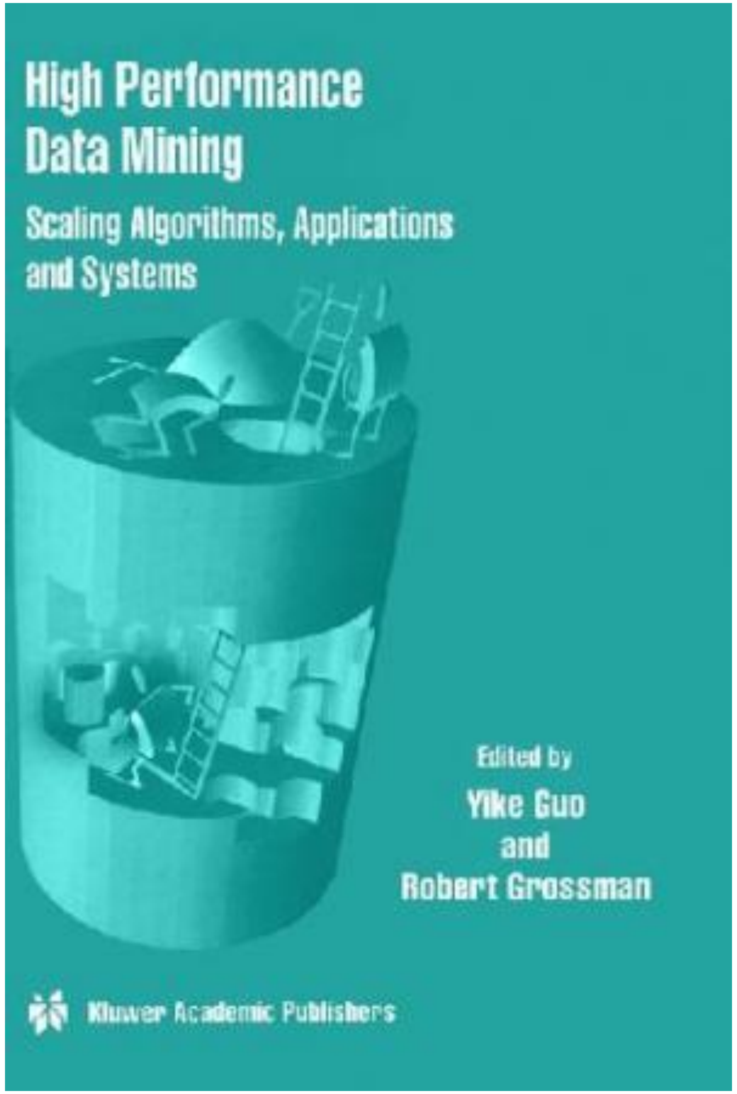
TEXTS IN COMPUTATIONAL SCIENCE AND ENGINEERING 13

Mark H. Holmes

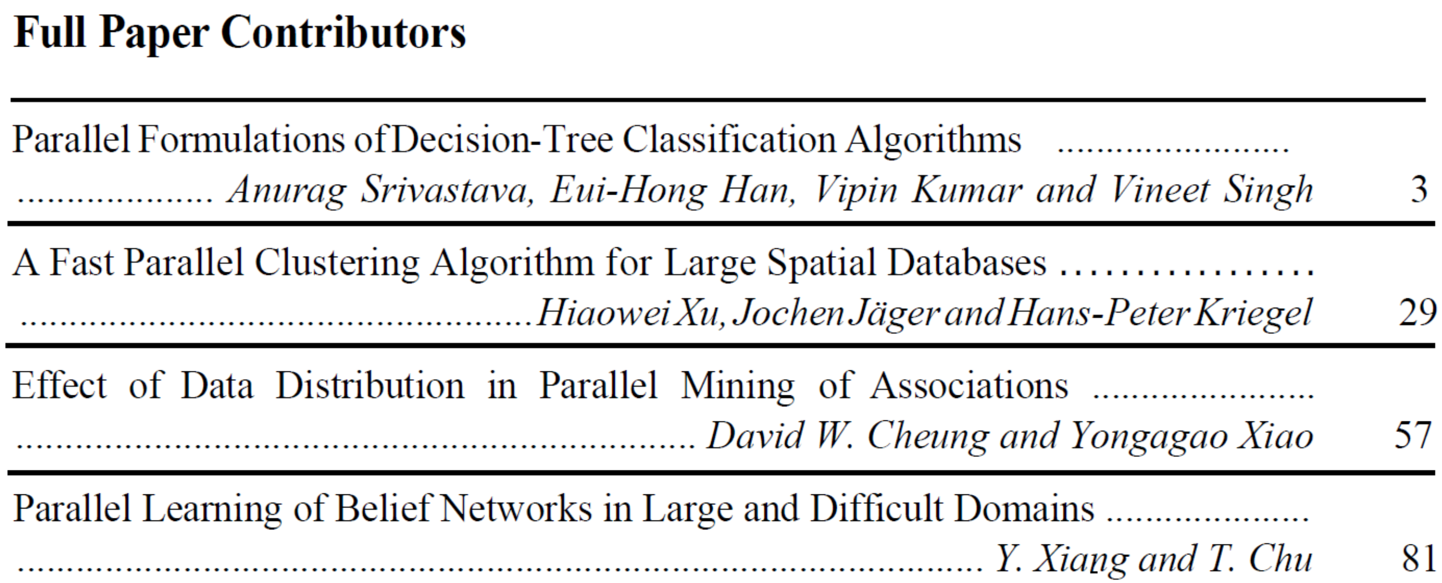
Introduction to Scientific Computing and Data Analysis

Springer

- □ Introduction to Scientific Computing and Data Analysis
- □ Authors Mark H. Holmes (auth.)
- □ Year 2016
- □ Pages 505
- □ Publisher Springer International Publishing
- □ Language en
- □ ISBN 9783319302546



- High Performance Data Mining: Scaling Algorithms, Applications and Systems
- Yike Guo, Robert Grossman
- **2000**



Full Paper Contributors