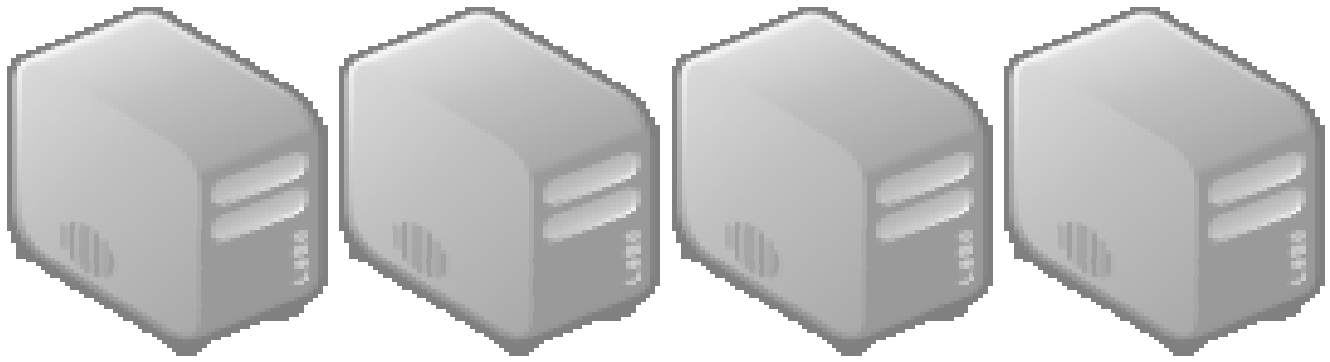


高速運算演進史

各種電腦運算型態及其優缺點

Jazz Wang
Yao-Tsung Wang
jazz@nchc.org.tw

PART 1 : (60 %)

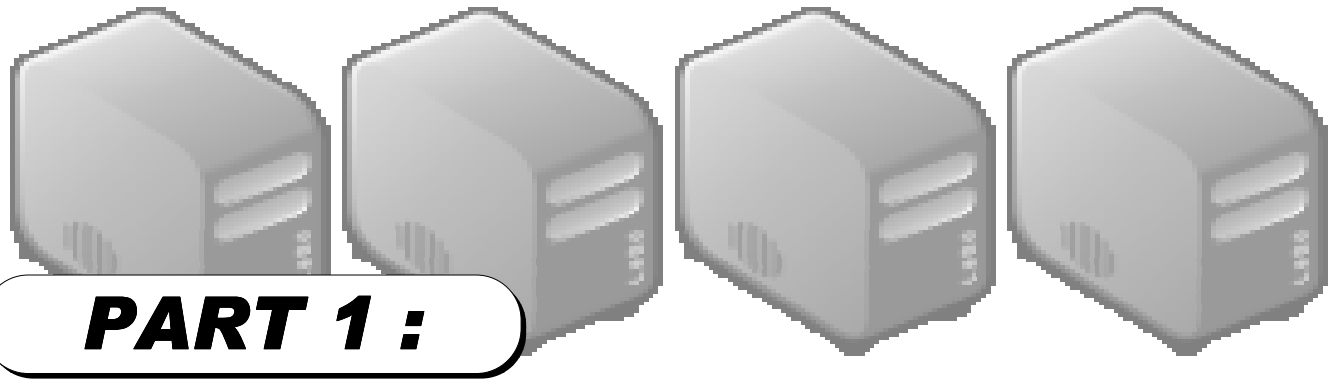
HPC = High Performance Computing
What is HPC? Types of HPC ?
Can I solve my problem with HPC ?

PART 2 : (30%)

HPC & Bioinformatics Application

PART 3 : (10%)

Open Source for Bioinformatics



PART 1 :

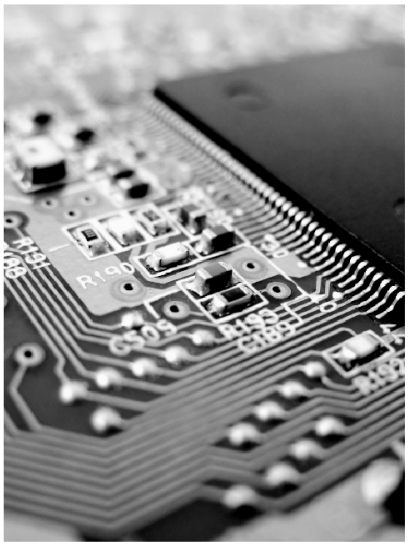
HPC 101

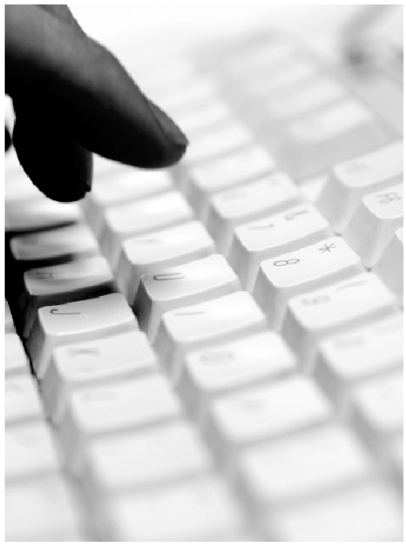
Jazz Wang
Yao-Tsung Wang
jazz@nchc.org.tw

What is HPC ?

&

Why HPC ?





High performance computing brings together computers, software, and expertise to solve problems too difficult to solve effectively by other means.

Source: http://insidehpc.com/whatishpc/WhatIsHPC.pdf

Simulated crash tests improve auto safety in more scenarios than could be tested in the real world

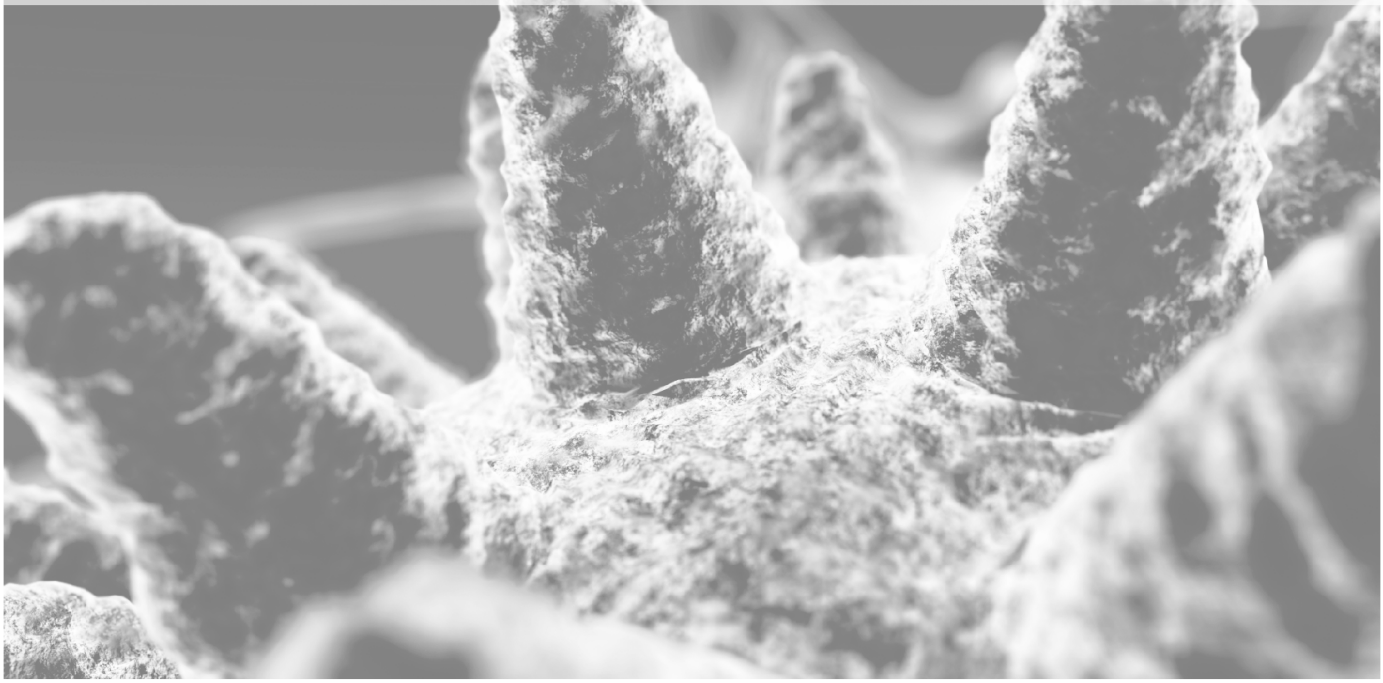
Too dangerous

Source: http://insidehpc.com/whatishpc/WhatIsHPC.pdf

Source: http://insidehpc.com/whatishpc/WhatIsHPC.pdf

Source: http://insidehpc.com/whatishpc/WhatIsHPC.pdf

HPC gives doctors and scientists a window into the chemistry of our bodies to help develop new drugs and treatments



Too far beyond other tools

Why HPC?

www.insideHPC.com/HPCcan

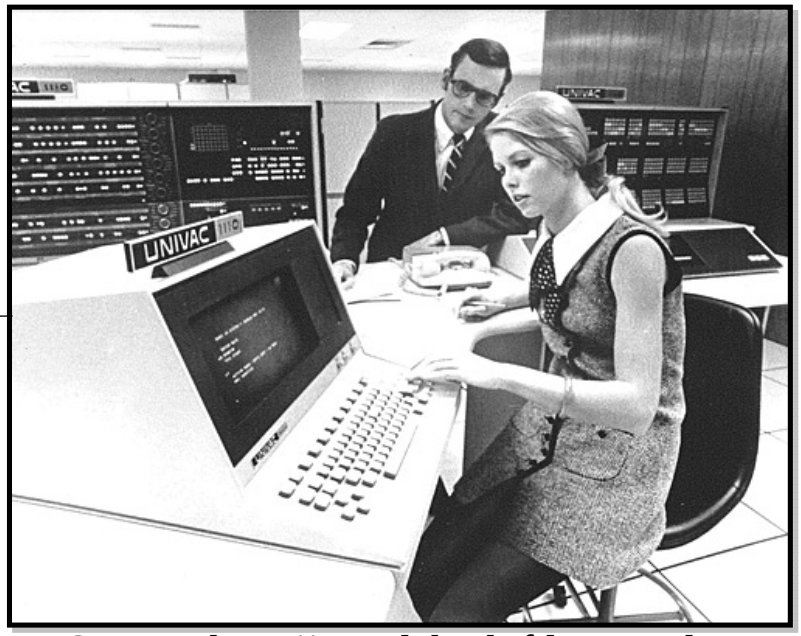
Source: http://insidehpc.com/whatishpc/WhatIsHPC.pdf

Types of HPC ?

Source: http://blog.tice.de/a_icons/icons/512%20Time%20Machine.png

Back to Year 1960s ...

Brief History of Computing (1/5)



Source: http://pinedakrch.files.wordpress.com/2007/07/

1960 PDP-1

.

.

.

1965 PDP-7

.

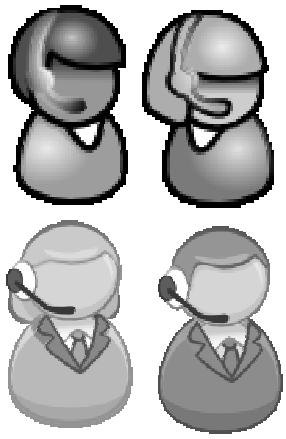
.

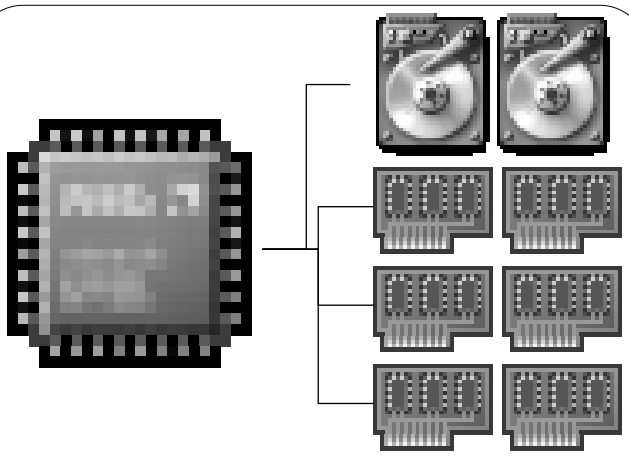
.

1969 1st Unix

Mainframe Super Computer

Evolution of Computing Architecture (1/5)





Multiple Users

Single CPU

Shared Memory

One Admin.

Mainframe Super Computer

Single Super Computer

使用者心裡的『謎之聲』(1/5)

等執行程式，要排隊排好久喔~

可惡，程式又死掉了，又得重排一次

真希望自己有一台電腦可以跑!!

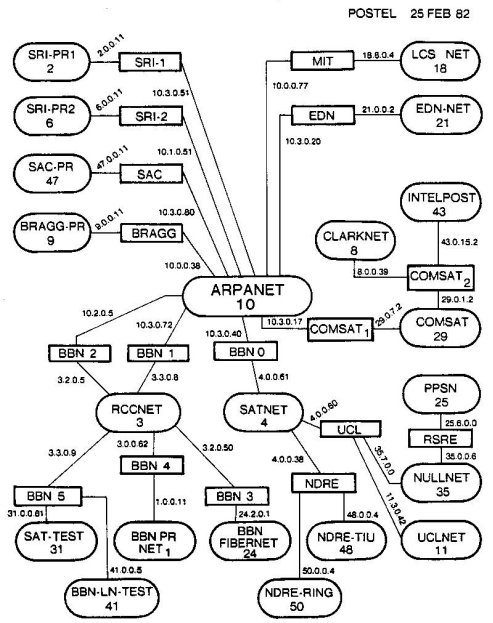
超級電腦是有錢人才玩得起的玩具~

1977 Apple II

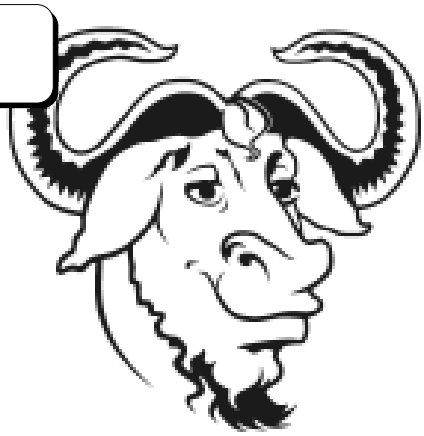
1981 IBM 1st PC 5150

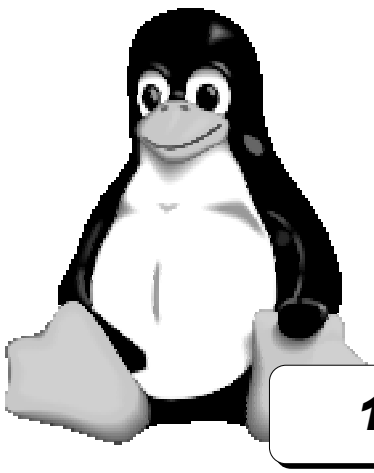
Back to Year 1970s ...

1982 TCP/IP



1983 GNU





1991 Linux

Back to Year 1980s ...

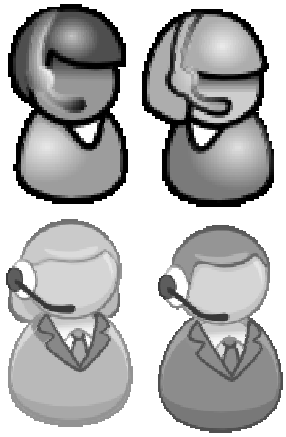
Brief History of Computing (2/5)

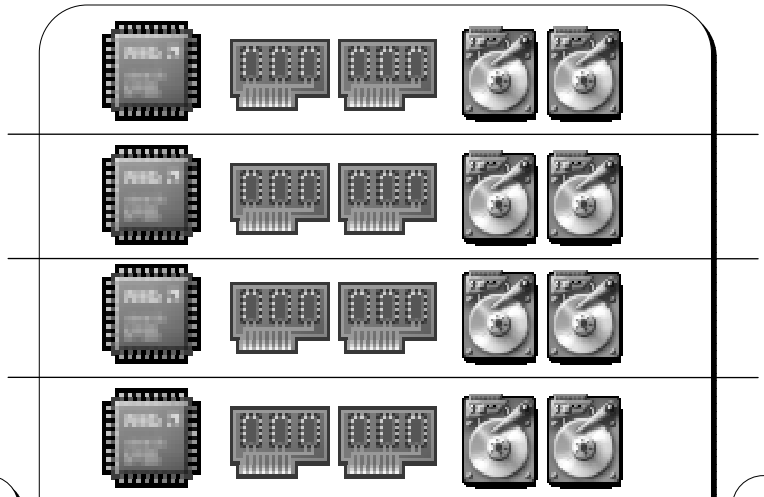
Source: http://www.nchc.org.tw

Mainframe Super Computer

PC / Linux Cluster Parallel

Evolution of Computing Architecture (2/5)





Multiple Users

Separate CPU

Separate Memory

One Admin.

ame r ter

PC / Linux Cluster Parallel

Multiple PC in One Location

使用者心裡的『謎之聲』(2/5)

可惡，記憶體不夠大，程式又死掉了

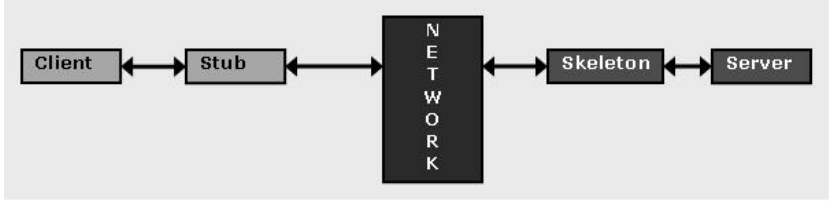
奇怪，我的程式為什麼不能跑？

真希望自己有一組叢集可以跑!!

管理員老大，可以幫我裝LiBT嗎？

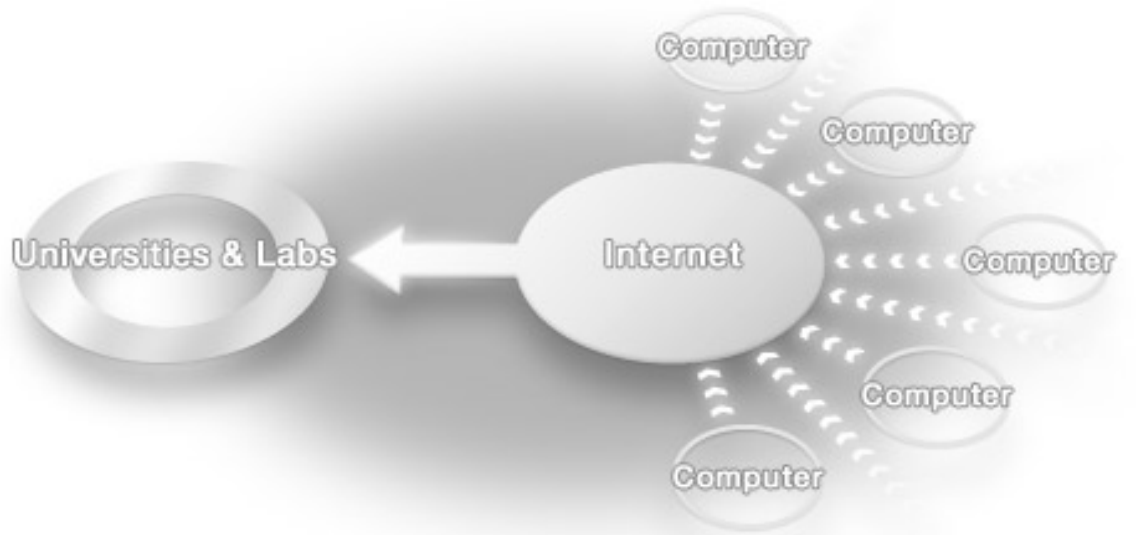
1990 World Wide Web by CERN
...
...
1993 Web Browser Mosaic by NCSA

1991 CORBA
...
Java RMI
Microsoft DCOM
...
Distributed Objects



Back to Year 1990s ...

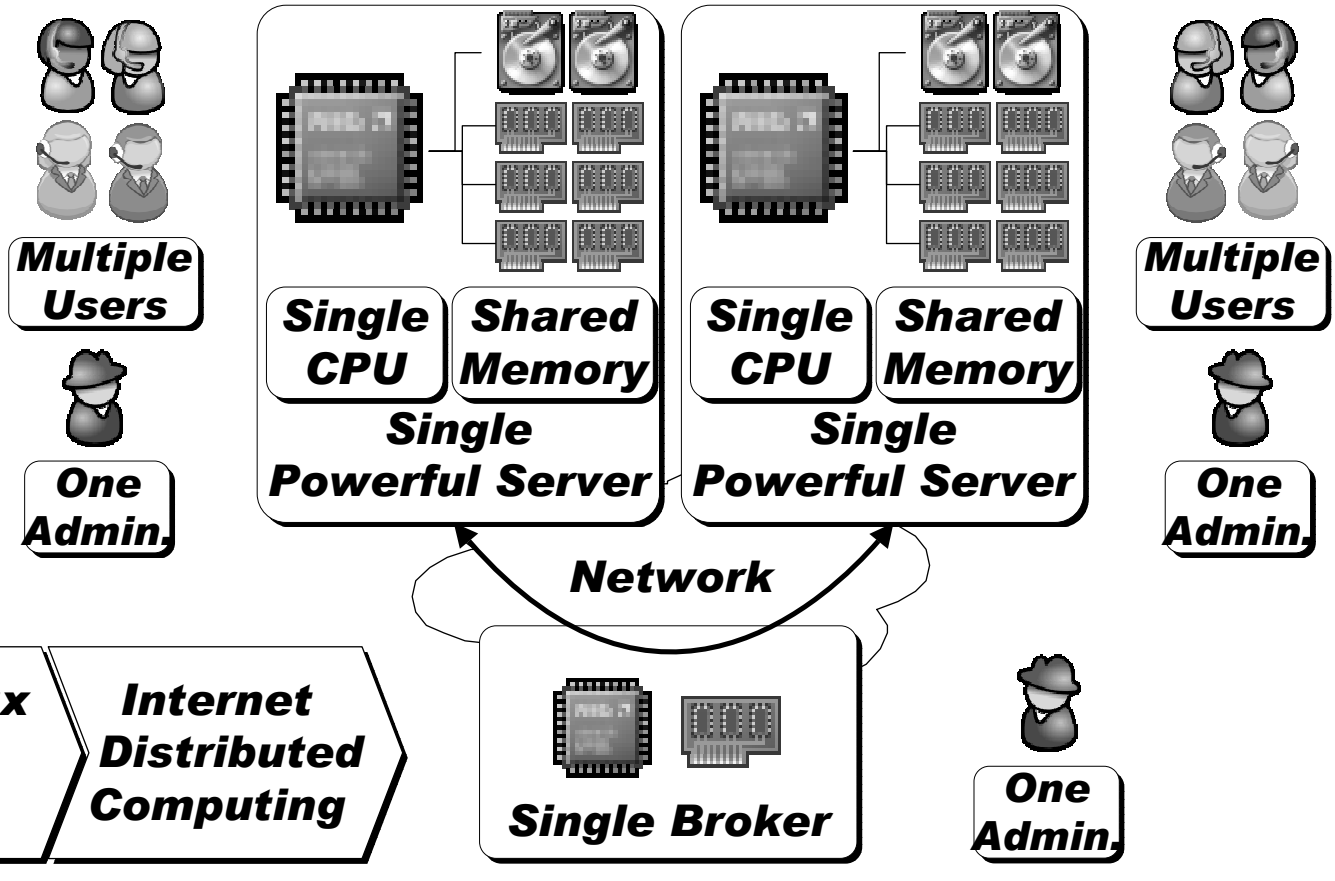
Brief History of Computing (3/5)



Source: http://www.scei.co.jp/folding/en/dc.html

Mainframe Super Computer → **PC / Linux Cluster Parallel** → **Internet Distributed Computing**

Evolution of Computing Architecture (3/5)



使用者心裡的『謎之聲』(3/5)

分散式物件怎麼這麼抽象啊～ＸＤ

啊！網路斷線了～不能動了～

大家把閒置電腦都貢獻出來吧!!

給我網路遊戲，其餘免談！

1997 Volunteer Computing
1999 SETI@HOME

2003 Globus Toolkit 2

2002 Berkley BOINC

2004 EGEE gLite

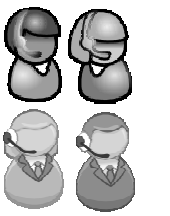
Back to Year 2000s ...

Brief History of Computing (4/5)

Source: http://gridcafe.web.cern.ch/gridcafe/whatisgrid/whatis.html

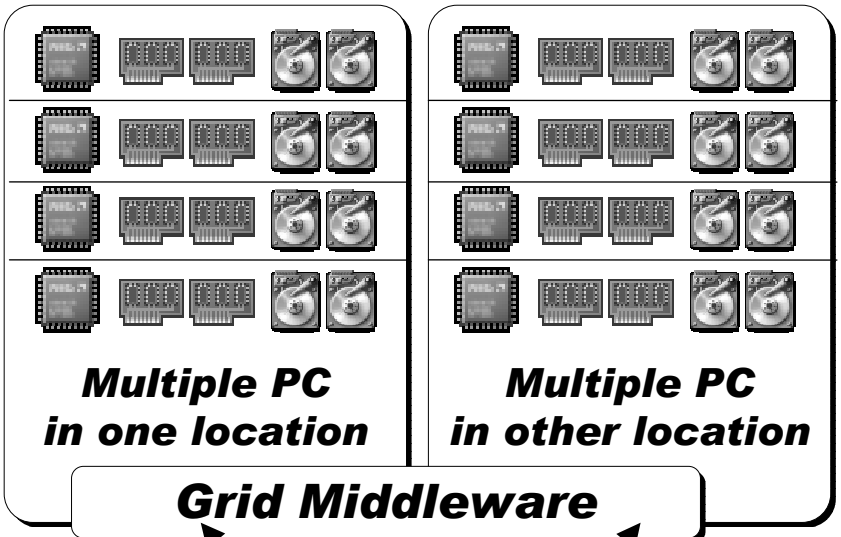
Mainframe Super Computer → PC / Linux Cluster Parallel → Internet Distributed Computing → Virtual Org. Grid Computing

Evolution of Computing Architecture (4/5)



Multiple Users

One Admin.

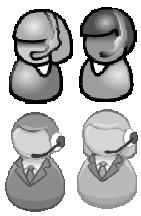


Multiple PC in one location

Multiple PC in other location

Grid Middleware

Network



Multiple Users

One Admin.

ernet tributed puting → Virtual Org. Grid Computing

Virtual Organization

Heterogeneous

CyberInfrastructure

使用者心裡的『謎之聲』(4/5)

已給我認證了，為什麼要不到資源？

啥？可用資源在美國，慢慢搬檔案吧！

為什麼人家Google那麼會算?!

長官，請幫我們去談好資源共享政策吧！

2001 Autonomic Computing
IBM

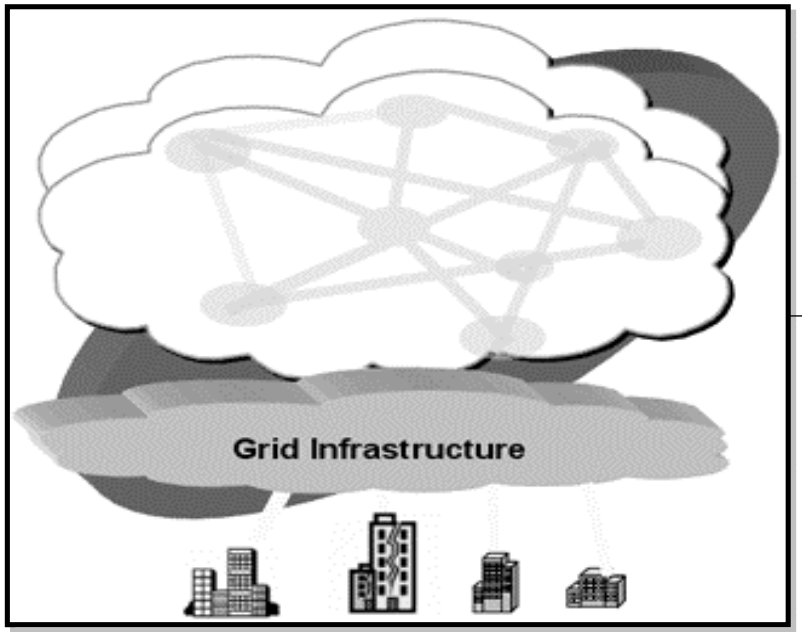
2006 Apache Hadoop

2005 Utility Computing
Amazon EC2 / S3

2007 Cloud Computing
Google + IBM

Back to Year 2007 ...

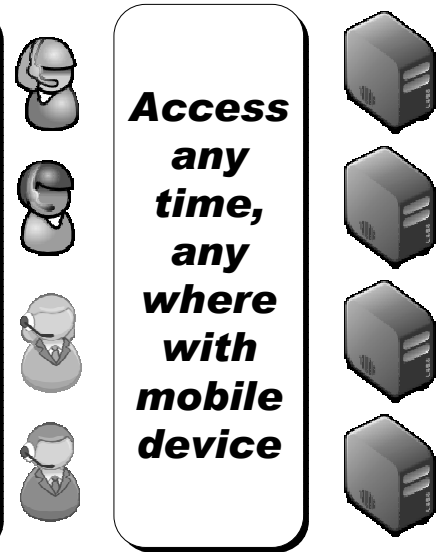
Brief History of Computing (5/5)



Source: http://mmdays.com/2008/02/14/cloud-computing/

...frame ...per ...puter → PC / Linux Cluster Parallel → Internet Distributed Computing → Virtual Org. Grid Computing → Data Explode Cloud Computing

Evolution of Computing Architecture (5/5)

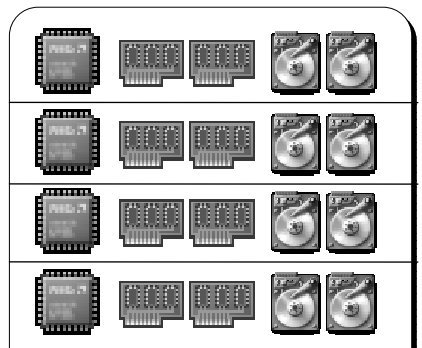


Each User

||

Virtual Admin.

Access any time, any where with mobile device



Multiple PC in different locations

Multiple Admin.

Virtual World | Physical World

...ual Org. ... Grid ...mputing → Data Explode Cloud Computing

What is NEXT ?!
Mobile Computing ?!

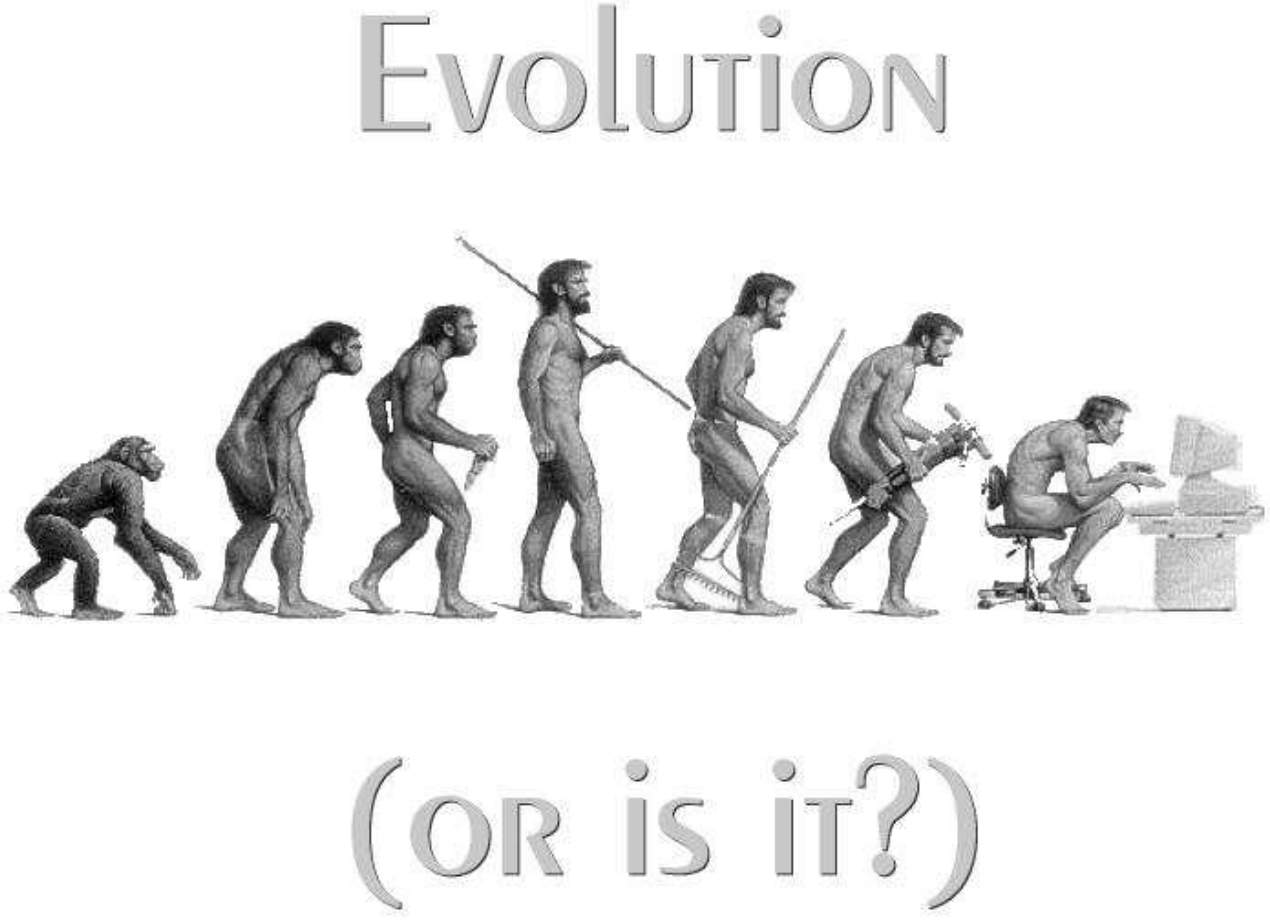
使用者心裡的『謎之聲』(5/5)

雲端運算合適我用嗎？

按使用時間計費，真的比較省？

Google到底有沒有偷窺我的信?!

我們自己可以架雲端運算的環境嗎？



Source: http://cyberpingui.free.fr/humour/evolution-white.jpg

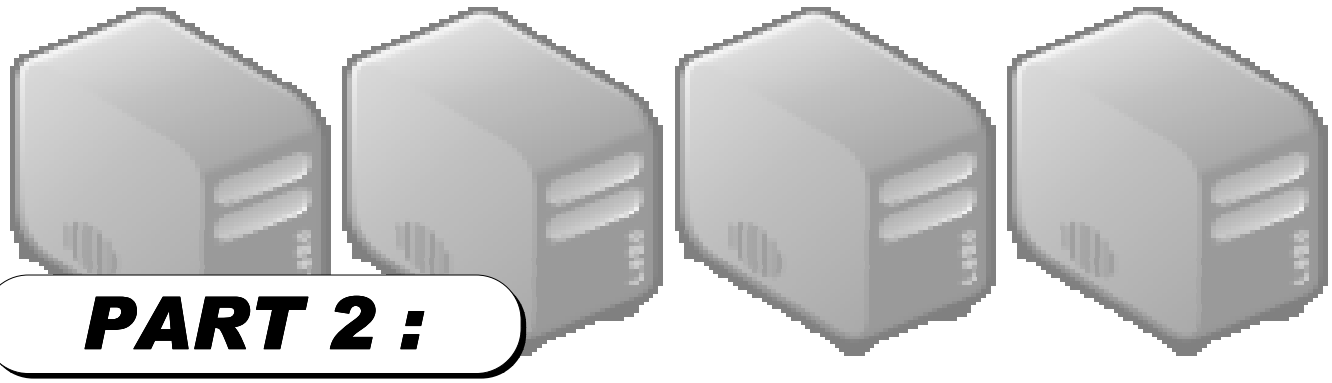
Flying to the Cloud ...
or
Falling to the Ground ...

Source: http://media.photobucket.com/image/falling%20ground/preeto_f10/falling.jpg

Which Type of HPC is the Right ONE to solve My Problem ?

不負責解析

	執行程式 記憶體需求	執行程式 邏輯可分割	輸入資料 數量/大小	應用特性 計算特性
超級電腦 ***Mainframe***	非常大 ***(比PC大)***	不易分割	單一大檔 ***GB***	即時性高 共享記憶體
叢集運算***Cluste*** ***Parallel***	小於單一 計算節點 配置量	邏輯相近 可分割	一般數量 ***MB***	即時性一般 共享檔案
分散式運算 ***Distributed***	小於單一 計算節點 配置量	邏輯不同 偕同運作	一般數量 ***MB***	即時性高 訊息傳遞
格網運算 ***Grid***	小於單一 計算節點 配置量	邏輯相近 可分割	一般數量 ***MB***	即時性較低 資料就計算
雲端運算 ***Cloud***	小於單一 計算節點 配置量	邏輯相近 各自獨立	海量檔案 ***MB***	即時性低 計算就資料



PART 2 :

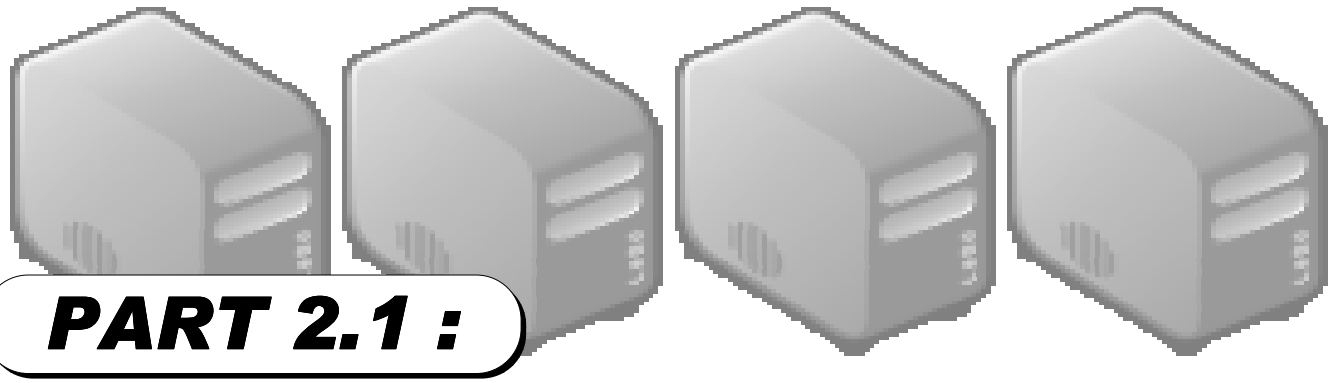
HPC & Bioinformatics Application

Jazz Wang
Yao-Tsung Wang
jazz@nchc.org.tw

BLAST (Basic Local Alignment Search Tool)

- **http://blast.ncbi.nlm.nih.gov/**
- **National Center for Biotechnology Information**
- **BLAST is an algorithm for comparing primary biological sequence information.** (BLAST用來比對生物序列的主要結構)
  - **the amino-acid sequences of different proteins**
  - **the nucleotides of DNA sequences**
  - (例如：不同蛋白質的氨基酸序列DNA序列的核甘酸)
- 用途：搜尋其他物種(如：老鼠)未知基因，是否也存在人類基因中
- 優點：使用啟發式搜索來找出相關的序列，比動態規劃快上**50**倍。
- 缺點：不能夠保證搜尋到的序列和所要找的序列之間的相關性。
- 技術問題：巨大的序列資料庫需要進行比對，怎樣計算才快？

- **Source: http://zh.wikipedia.org/w/index.php?title=BLAST_(生物資訊學)&variant=zh-tw**



PART 2.1 :

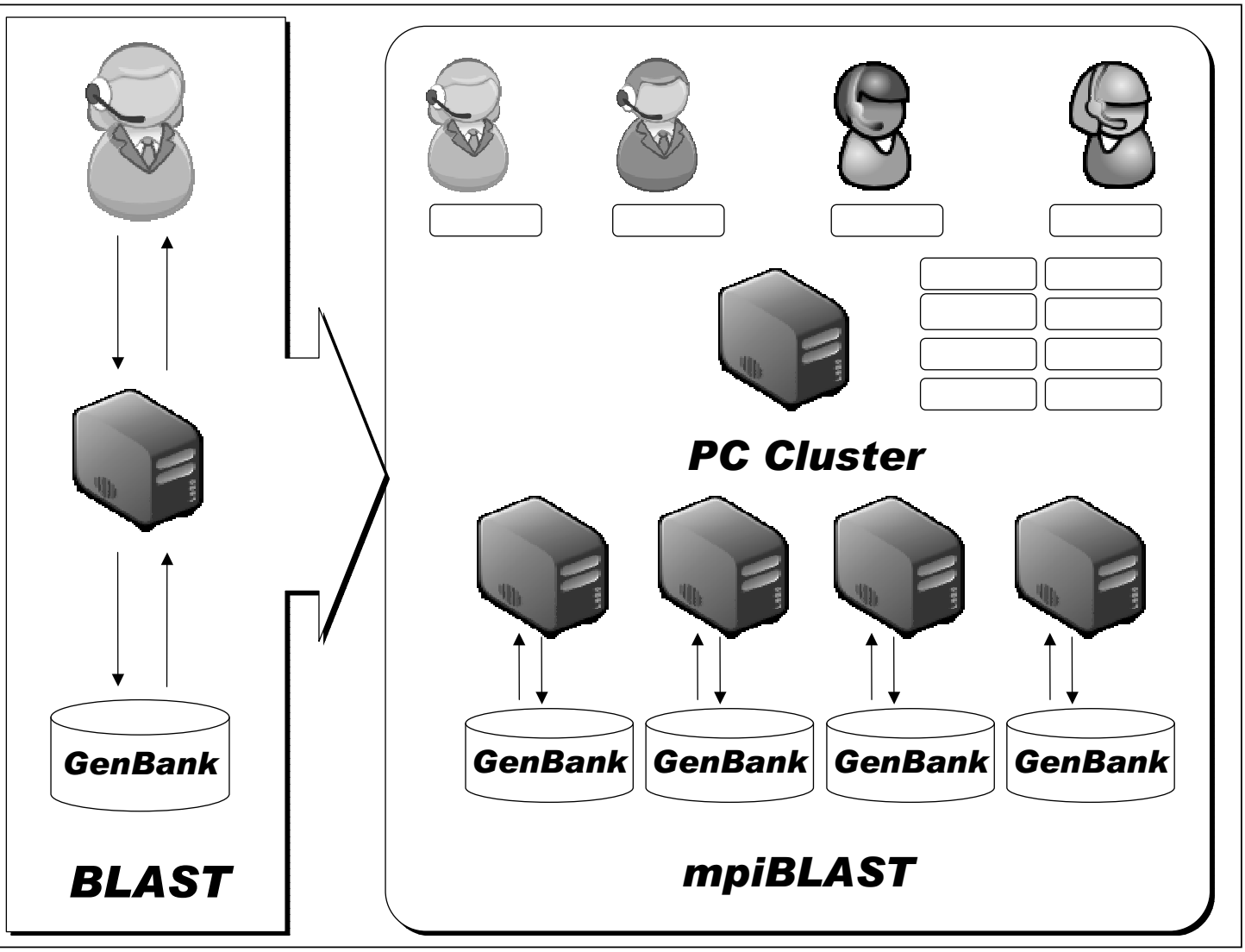
Cluster 101 & mpiBLAST

Jazz Wang
Yao-Tsung Wang
jazz@nchc.org.tw

Powered by DRBL

mpiBLAST

- ***http://www.mpiblast.org/***
- ***An open-source, parallel implementation of NCBI BLAST***
- 特點：
  - ***Database fragmentation***
  - ***Query segmentation***
  - ***Parallel input/output***
- 設計理念：
  - ***The Design, Implementation, and Evaluation of mpiBLAST.***
  - ***http://www.mpiblast.org/downloads/pubs/cwce03.pdf***
- 類似工具：
  - ***TurboWorx TurboBLAST***
  - ***Parallel BLAST by Caltech***



PART 2.2 :

Grid 101 & mpiBLAST-G2

Jazz Wang
Yao-Tsung Wang
jazz@nchc.org.tw

Powered by DRBL

mpiBLAST-G2

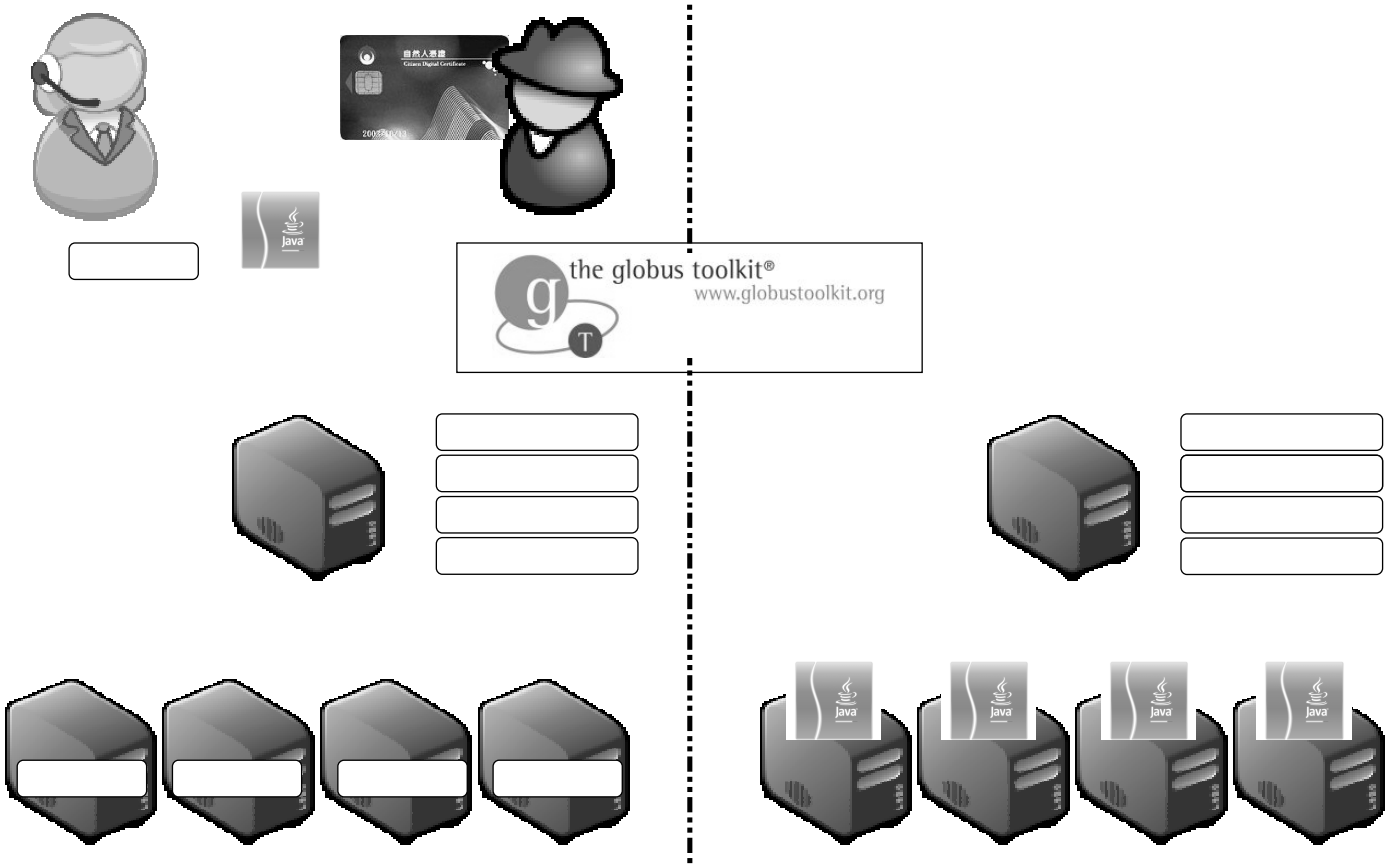
- ***mpiBLAST-G2 is an enhanced parallel program of LANL's mpiBLAST. It is based on Globus Toolkit 2.x and MPICH-g2.***
- ***Bioinformatics Technology and Service (BITS) team of Academia Sinica Computing Centre (ASCC), Taiwan***
- 參考：
  - ***The MPIBLAST-g2 Introduction***
  - ***MPIBLAST-g2 Example***
  - ***mpiBlast-G2 with GT4***

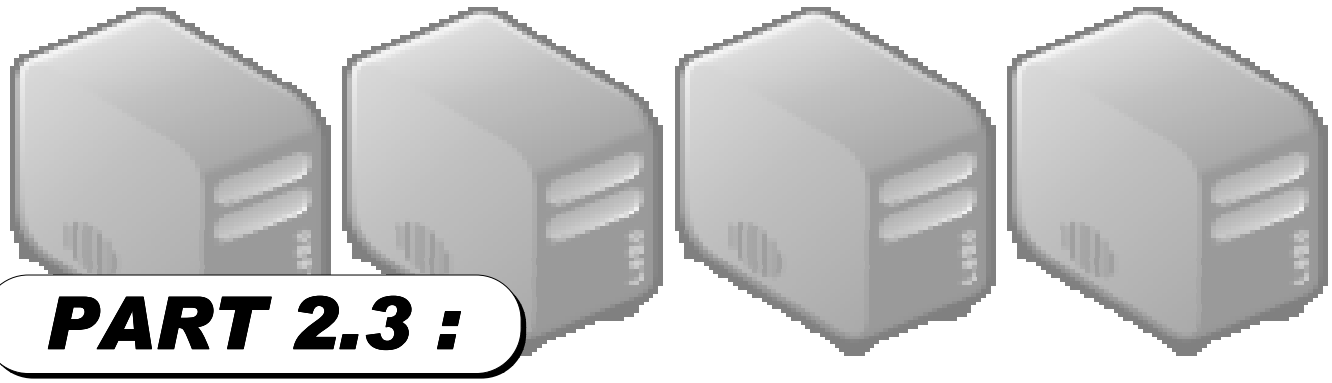
mpiBLAST

the globus toolkit®
www.globustoolkit.org

ASCC
ACADEMIA SINICA
Computing Center
中央研究院計算中心

Grid =~ Cluster of Cluster





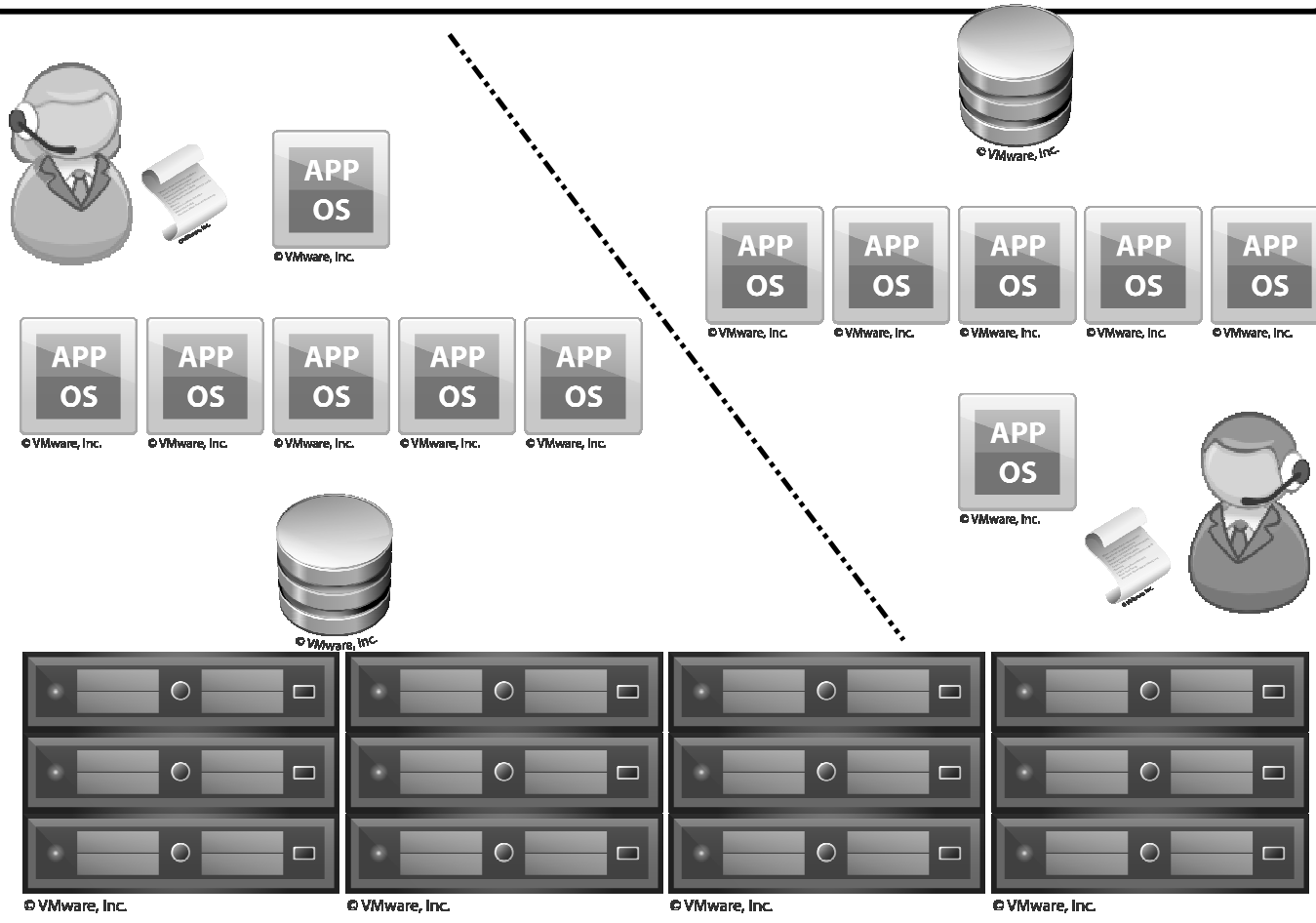
PART 2.3 :

Cloud 101 & CloudBLAST

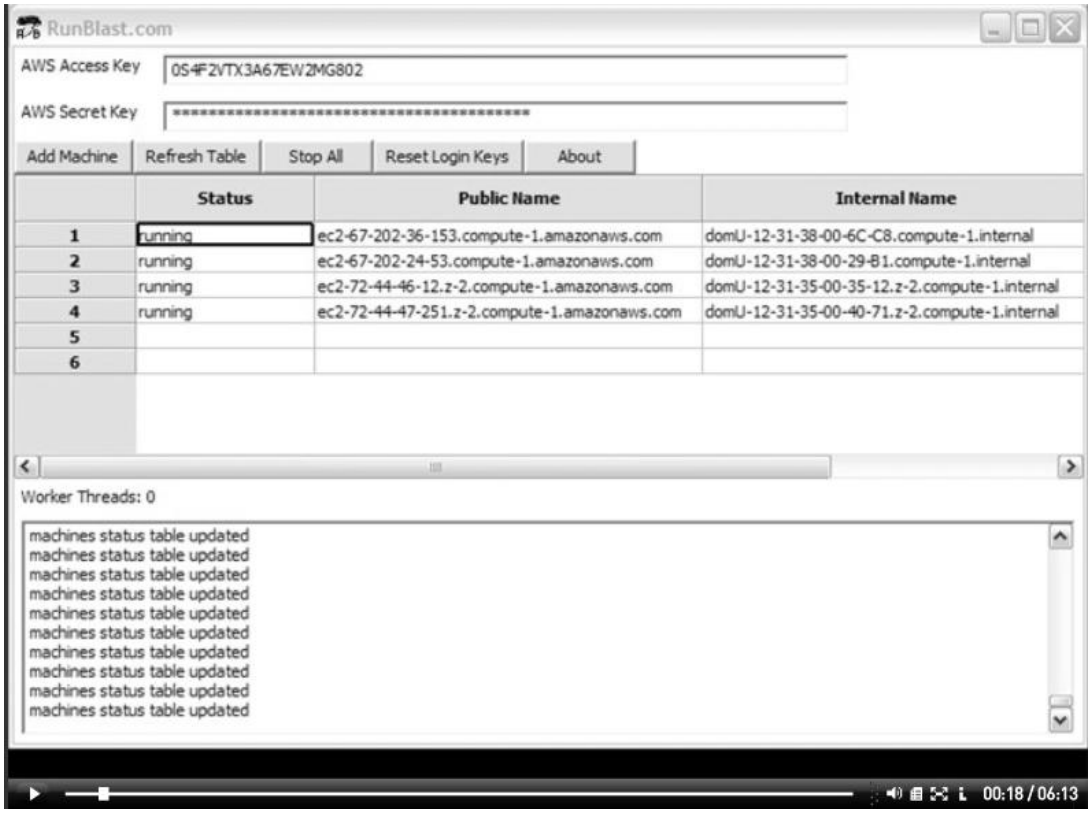
Jazz Wang
Yao-Tsung Wang
jazz@nchc.org.tw

Powered by **DRBL**

Cloud =~ Lots of Virtual Cluster

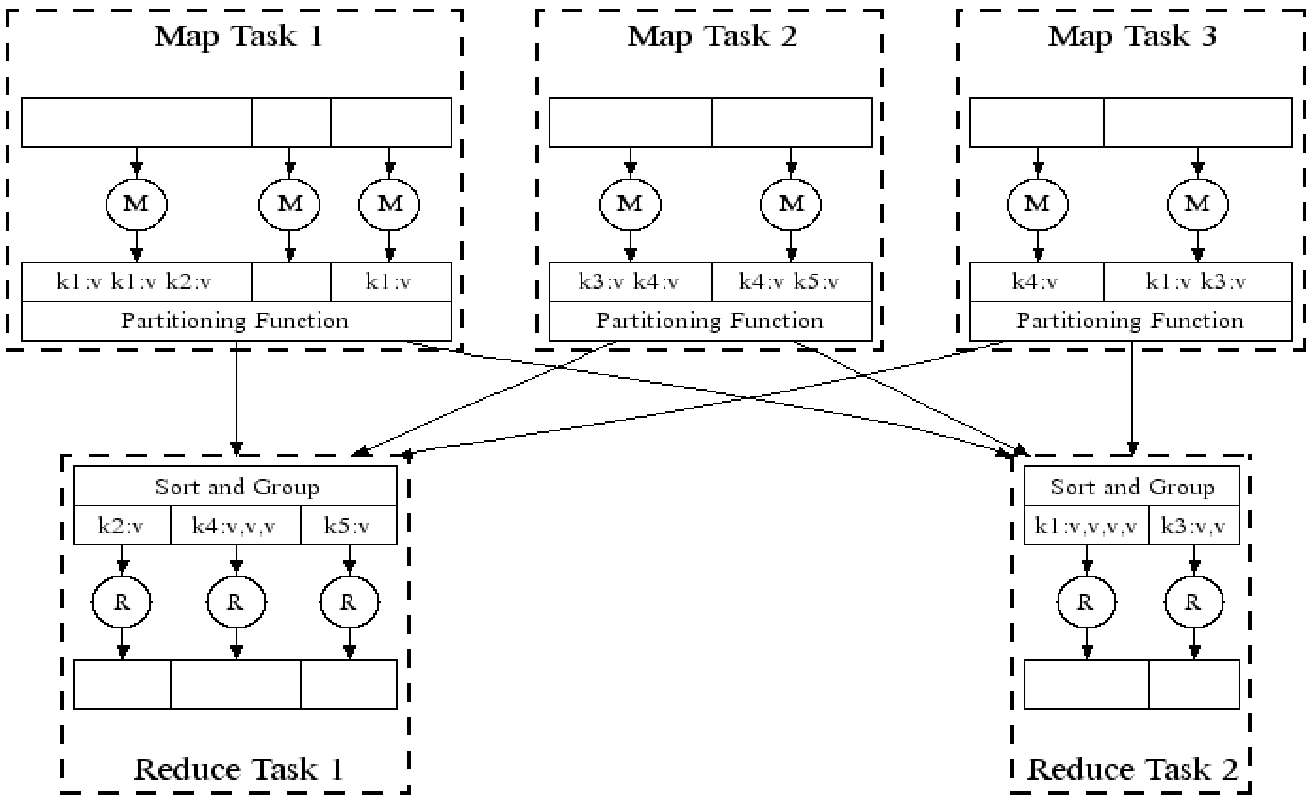


RunBLAST : mpiBLAST in Amazon EC2



Video: http://www.runblast.com/videos/runblast-blastwizard.swf

Map/Reduce



Ref. MapReduce: Simplified Data Processing on Large Clusters, Google

CloudBLAST

- ***"CloudBLAST: Combining MapReduce and Virtualization on Distributed Resources for Bioinformatics Applications", eScience 2008***
- 特點：採用***MapReduce***演算法進行***BLAST***運算

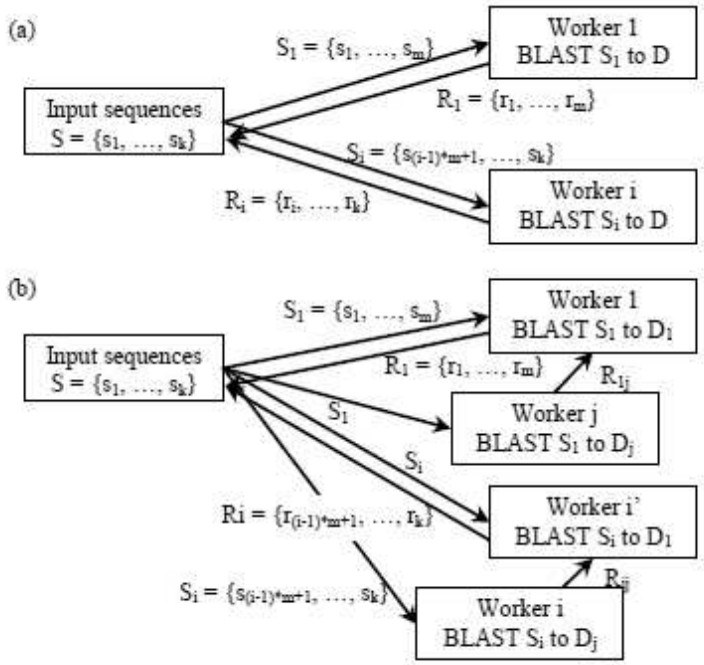
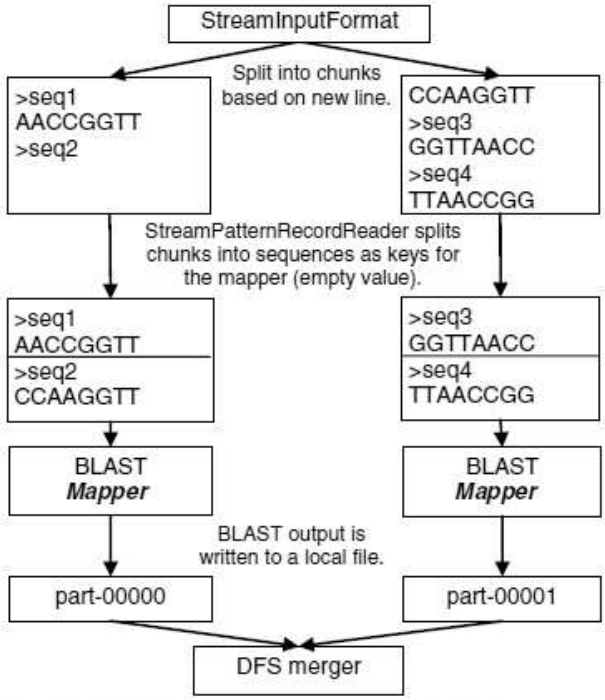
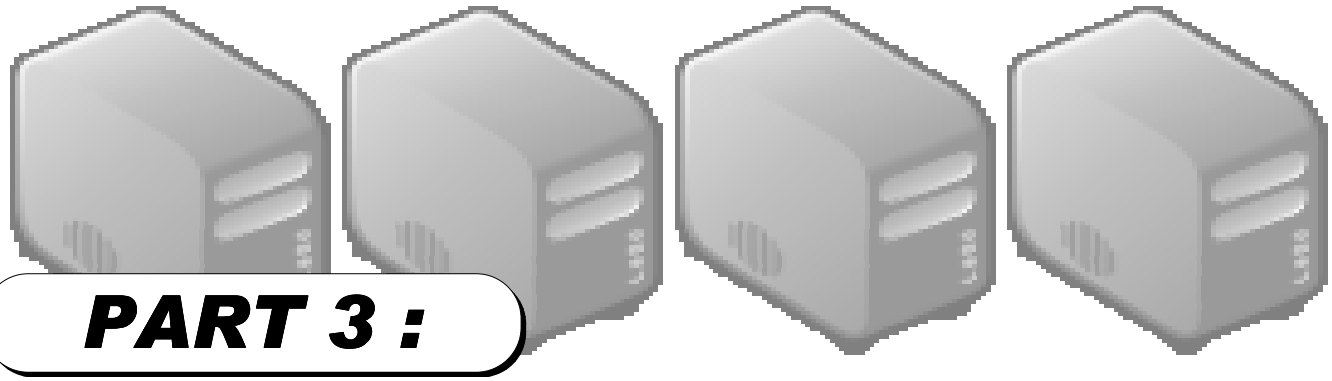


Figure 1. Parallelization of BLAST: (a) input sequences are partitioned





PART 3 :

Open Source for Bioinformatics

Jazz Wang
Yao-Tsung Wang
jazz@nchc.org.tw

Powered by **DRBL**

Open Source is your Friend !!

- Open Bioinformatics Foundation - http://www.bioinfomatics.org
  - BioPerl - http://bio.perl.org
  - BioPython - http://biopython.org
  - BioPHP - http://biophp.org
  - BioJava - http://biojava.org
- C++ Bio Sequence Library
  - http://libseq.sourceforge.net/
  - C++ 版本的序列分析函式庫
- Bio-SPICE - http://biospice.sourceforge.net/
- BioEra - http://bioera.net/
  - 跟腦科學有蠻強的關聯性，主要功能是在做訊號處理。
- NCBI Viewer - http://ncbiviewer.bravehost.com/

What we learn today ?

Q1: What is HPC? 何謂高速運算？
A1: HPC就是結合電腦硬體、軟體和一堆專家，
用各種方式解決困難的問題。

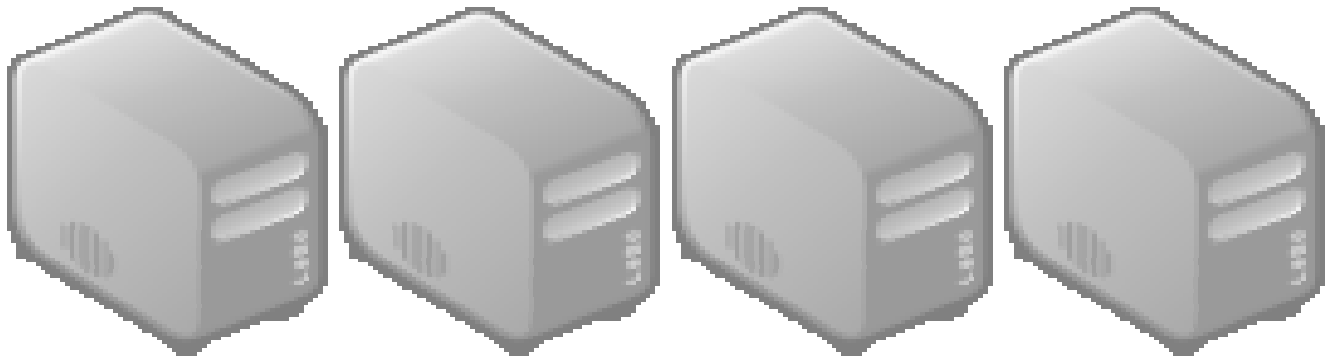
Q2: Types of HPC? 高速運算的種類有哪些？
A2: Mainframe, PC Cluster, Parallel, Distributed, Grid, Cloud
超級電腦、電腦叢集、平行、分散、格網、雲端運算

Q3: Can HPC solve all your problems? 高速運算可以解決所有問題***?***
A3: No. 高速運算無法解決所有問題，各種類別也各有所長。

Q4: What is PC Cluster? 何謂電腦叢集***?***
A4: Cluster = lots of PCs. 很多電腦用內部網路串起來，就是叢集。

Q5: What is Grid? 何謂格網運算***?***
A5: Grid = Cluster of Cluster. 把好幾座叢集視為一座抽象的叢集。

Q6: What is Cloud? 何謂雲端運算***?***
A6: Cloud = lots of Virtual Cluster. 在實體叢集中打造多座虛擬叢集。



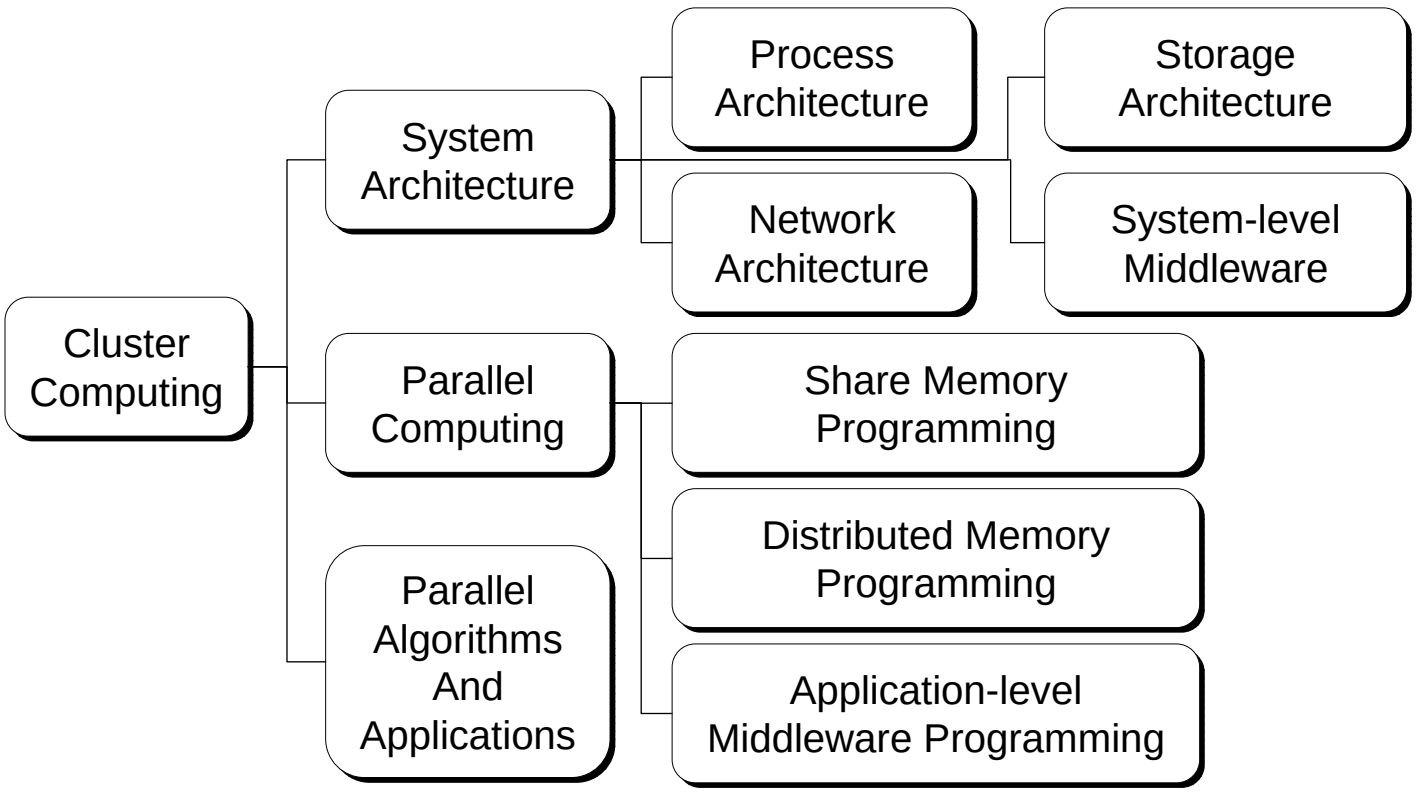
Questions?

Slides - http://bio.classcloud.org

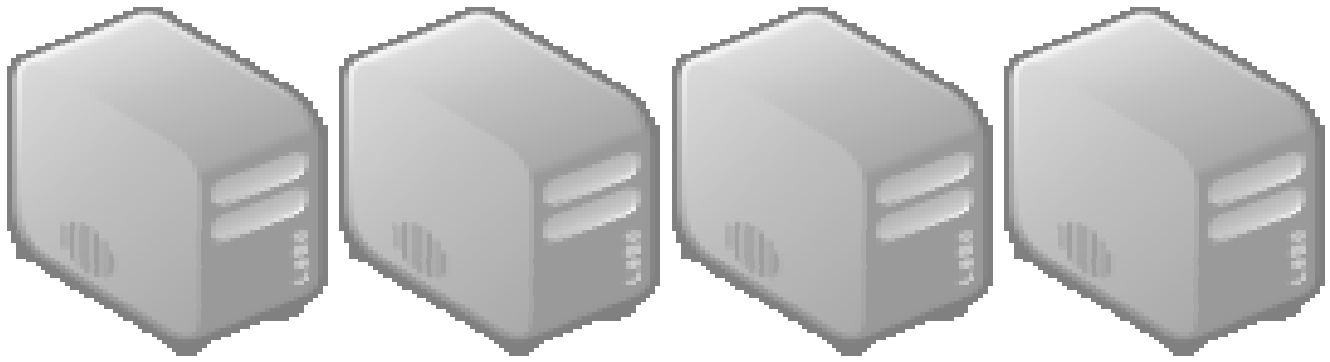
Jazz Wang
Yao-Tsung Wang
jazz@nchc.org.tw

Powered by DRBL

Research topics about PC Cluster

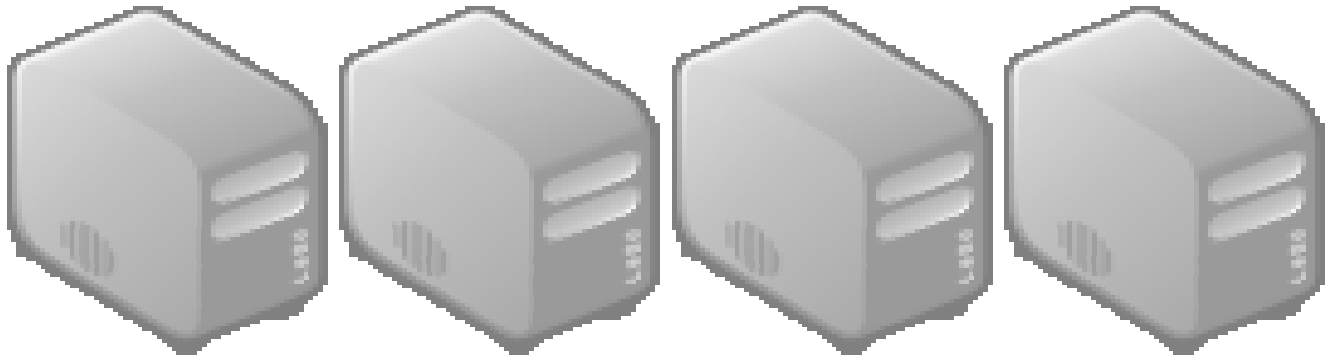


Ref: Cluster Computing in the Classroom: Topics, Guidelines, and Experiences
http://www.gridbus.org/papers/CC-Edu.pdf



叢集運算基本觀念

Jazz Wang
Yao-Tsung Wang
jazz@nchc.org.tw

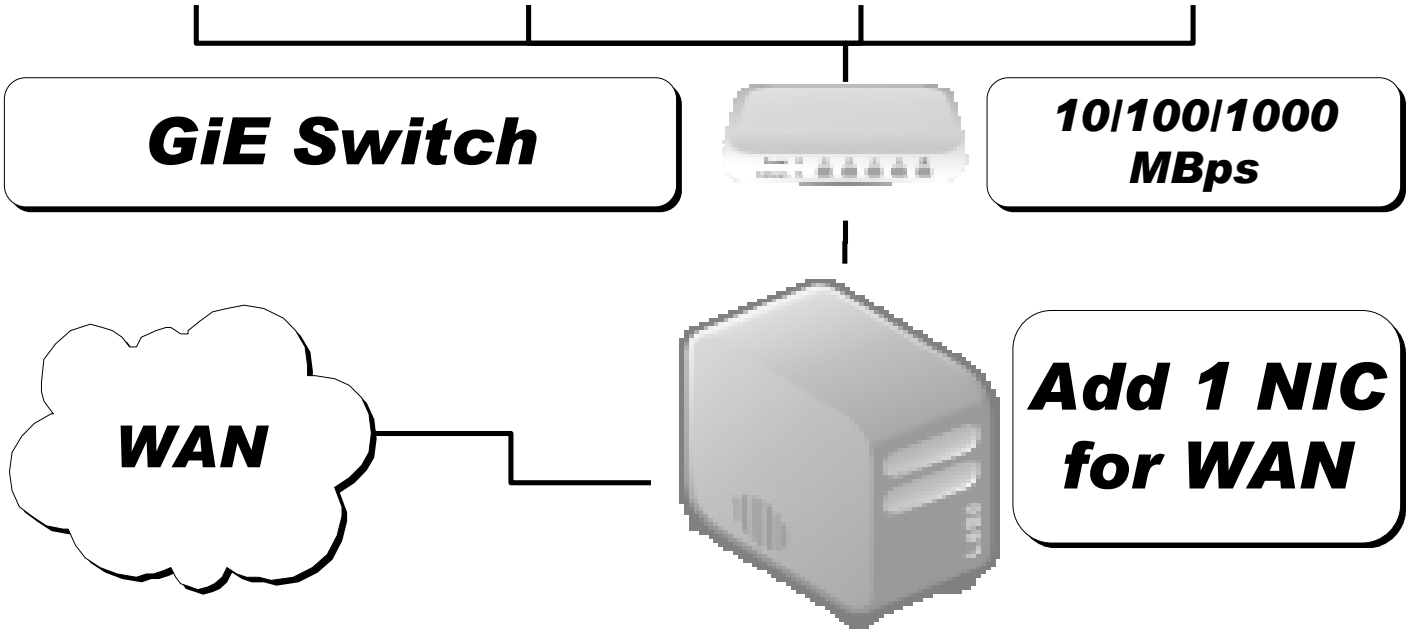


At First, We have “ 4 + 1 ” PC Cluster

It'd better be 2^n

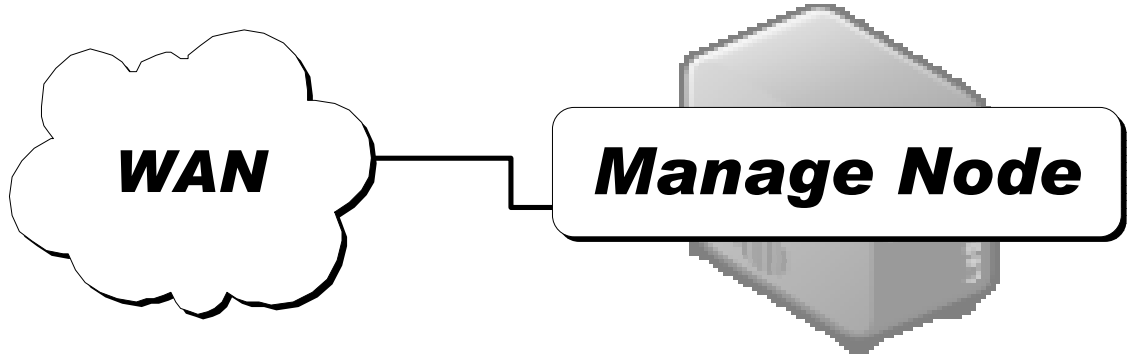
Manage Scheduler

Then, We connect 5 PCs with Gigabit Ethernet Switch



Compute Nodes

4 Compute Nodes will communicate via LAN Switch. Only Manage Node have Internet Access for Security!



Compute Nodes

Basic System Setup for Cluster

- Messaging
- Account Mgnt.
- MPICH
- SSHD
- NIS
- YP
- GCC
- GNU Libc
- Bash
- Perl
- Kernel Module
- Linux Kernel
- Boot Loader

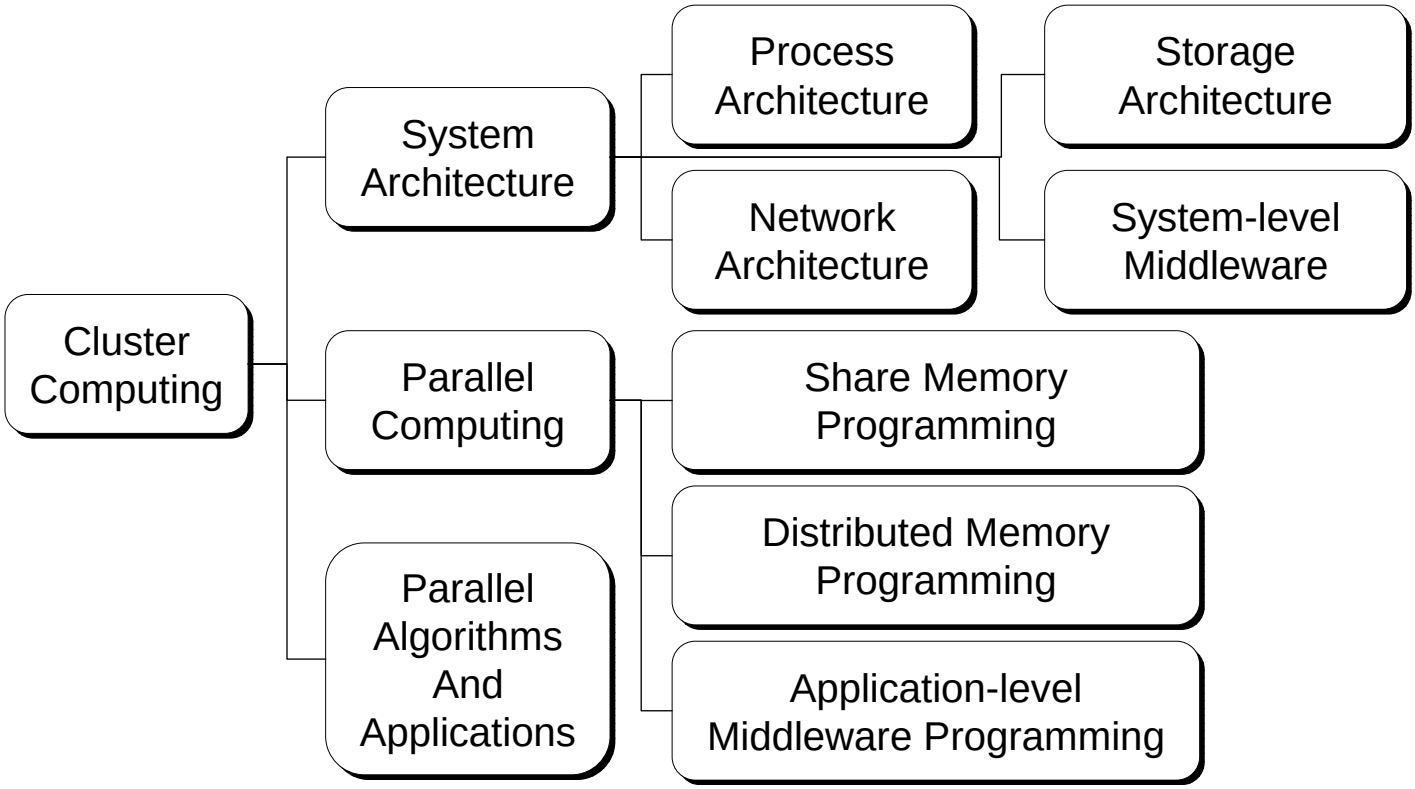
On Manage Node, We need to install Scheduler and Network File System for sharing Files with Compute Node

Extra

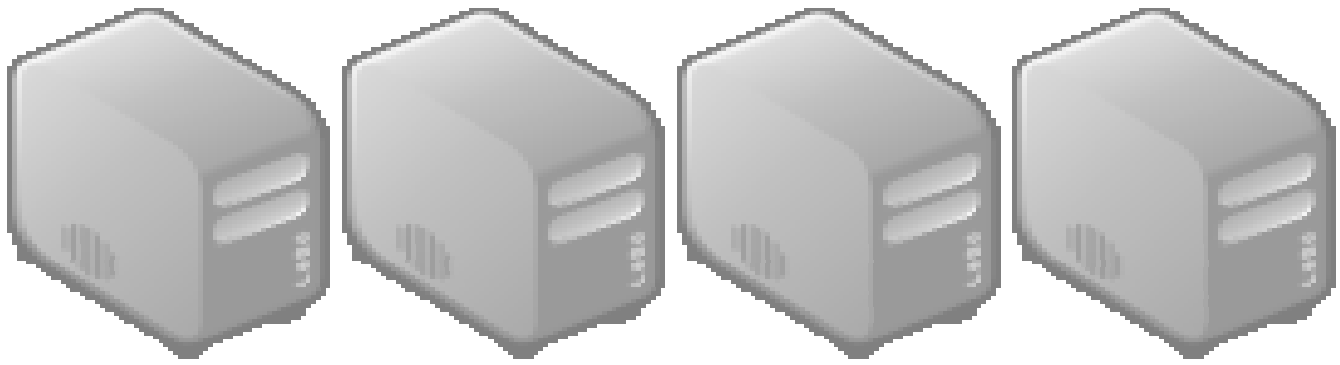
- Job Mgnt.
- OpenPBS
- File Sharing
- NFS

- Messaging
- Account Mgnt.
- MPICH
- SSHD
- NIS
- YP
- GCC
- GNU Libc
- Bash
- Perl
- Kernel Module
- Linux Kernel
- Boot Loader

Research topics about PC Cluster

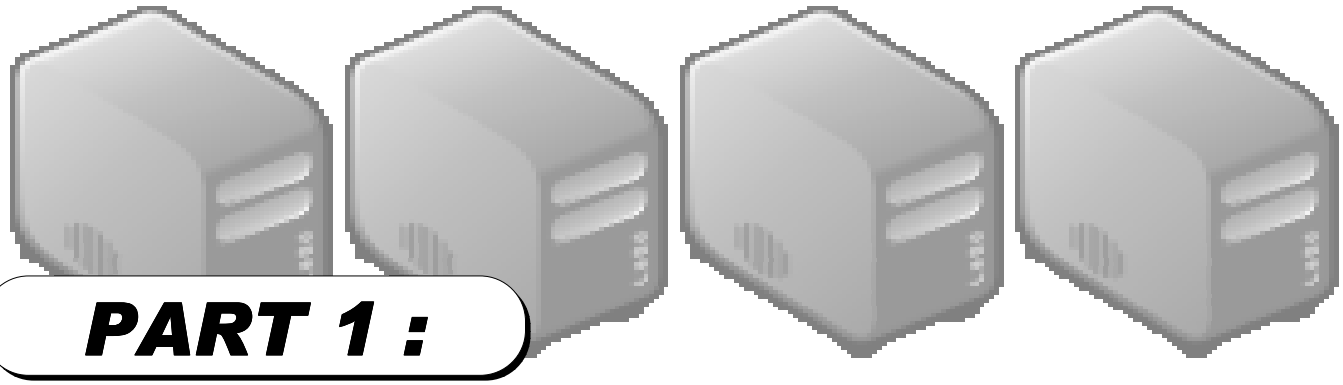


Ref: Cluster Computing in the Classroom: Topics, Guidelines, and Experiences
http://www.gridbus.org/papers/CC-Edu.pdf



企鵝龍與再生龍

Jazz Wang
Yao-Tsung Wang
jazz@nchc.org.tw



PART 1 :

開發背景簡介

Jazz Wang
Yao-Tsung Wang
jazz@nchc.org.tw

Stand On the Shoulder of Giants

自由軟體 (*Free Software*)

站在巨人的肩膀上，是自由軟體發展的理念。其靈活、可自由複製、分享的價值，將有效解決資訊教育的管理成本及商業軟體高成本負擔的問題。

降低資訊教育管理成本

需要「化繁為簡」的解決方案！

一般國內小學的電腦教室

☑ **人力、時間成本高**

教師1人維護管理多組設備

教學同時分派或收集作業

☑ **設備維護成本高**

需分別處理設定(每班約40台)

如：電腦中毒、環境設定

系統操作問題、開關機、

備份還原等

平衡商業軟體與知識教育

知識和軟體都需要讓孩子「帶著走」！

☑ **商業軟體授權高成本**

在校學習，也需回家複習

學校每台(平均) 2萬

學生家用(平均) 4萬

☑ **知識與法治的學習**

教育知識，也需教育尊重

尊重智財權觀念

國網中心自由軟體開發

多元化資訊教學的新選擇！

以個人叢集電腦(PC Cluster)經驗發展DRBL&Clonezilla

企鵝龍*DRBL*

(Diskless Remote Boot in Linux)

適合將整個電腦教室轉換成純自由軟體環境

再生龍*Clonezilla*

Clonezilla

適用完整系統備份、裸機還原或災難復原

是自由！不是免費...

分送、修改、存取、使用軟體的自由。免費是附加價值。

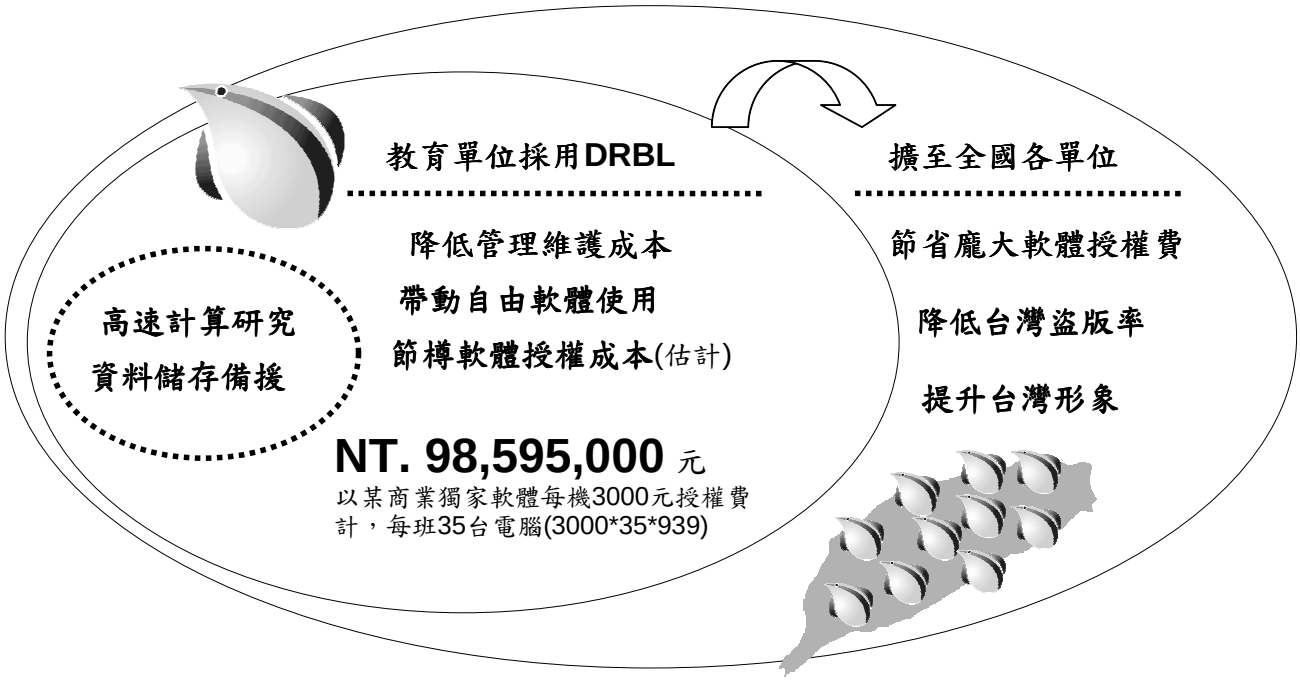
企鵝龍DRBL&再生龍Clonezilla

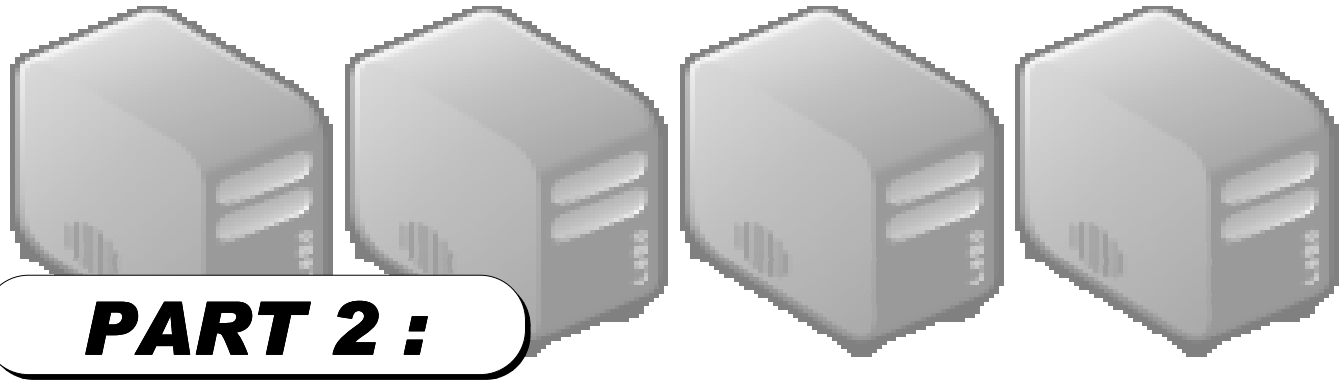
電腦教室管理的新利器！

■以每班40台電腦為估算單位

DRBL&Clonezilla	未使用	使　用
管理簡化	分別管理40台	管理 1台 伺服器
硬體設備成本	每台都需配備周邊硬體	伺服器控制，節約每台學生機之周邊硬體
軟體授權成本	40台:3000*40=120,000 (MS Windows授權1台電腦之授權費NT$3,000)	軟體授權 NT$0
合法複製、分享	需負擔授權費	複製合法 NT$0
多元化電腦教學	不同系統無法並存	Linux 與M$ Windows可並存

降低成本，提升形象





PART 2 :

企鵝龍的開機原理

Jazz Wang

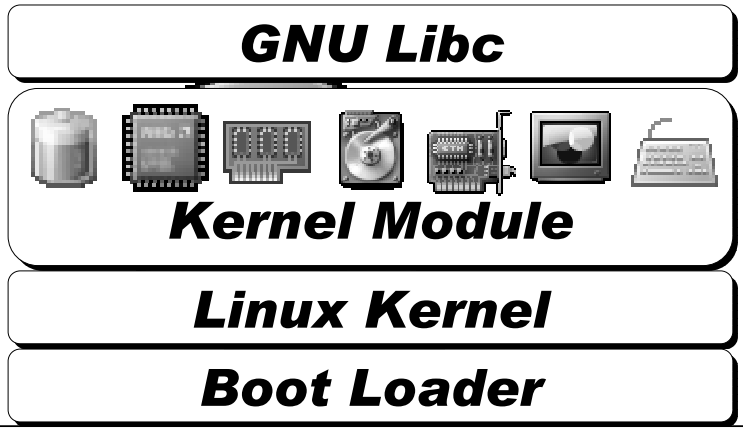
Yao-Tsung Wang

jazz@nchc.org.tw

1st, We install Base System of GNU/Linux on Management Node.

You can choose:

Redhat, Fedora, CentOS, Mandriva, Ubuntu, Debian, ...



2nd, We install DRBL package and configure it as DRBL Server. There are lots of service needed:

SSHD, DHCPD, TFTPD, NFS Server, NIS Server, YP Server ...

Network Booting			Account Mgnt.		
NFS	TFTPD	DHCPD	SSHD	NIS	YP
Perl	Bash	GNU Libc			

DRBL Server based on existing Open Source and keep Hacking!

Kernel Module

Linux Kernel

Boot Loader

After running "drblsrv -i" & "drblpush -i", there will be pxelinux, vmlinux-pex, initrd-pxe in TFTPROOT, and different configuration files for each Compute Node in NFSROOT

NFS	TFTPD	DHCPD	SSHD	NIS	YP

Config. Files Ex. hostname

initrd-pxe

vmlinuz-pxe

pxelinux

GNU Libc

Kernel Module

Linux Kernel

Boot Loader

3nd, We enable PXE function in BIOS configuration.

BIOS PXE | BIOS PXE | BIOS PXE | BIOS PXE

NFS | TFTPD | DHCPD | SSHD | NIS | YP

Config. Files
Ex. hostname

initrd-pxe

vmlinuz-pxe

pxelinux

GNU Libc

Kernel Module

Linux Kernel

Boot Loader

While Booting, PXE will query IP address from DHCPD.

BIOS PXE | BIOS PXE | BIOS PXE | BIOS PXE

NFS | TFTPD | DHCPD | SSHD | NIS | YP

Config. Files
Ex. hostname

initrd-pxe

vmlinuz-pxe

pxelinux

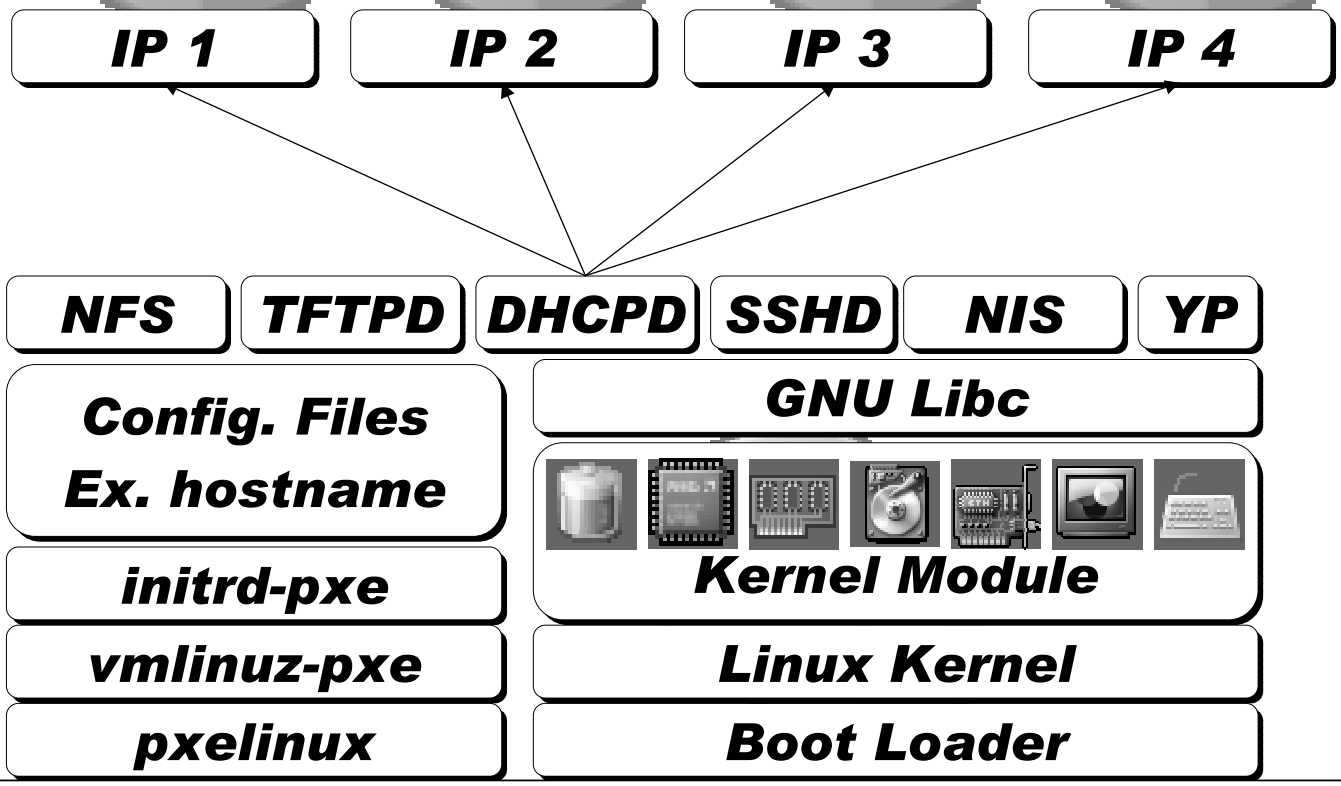
GNU Libc

Kernel Module

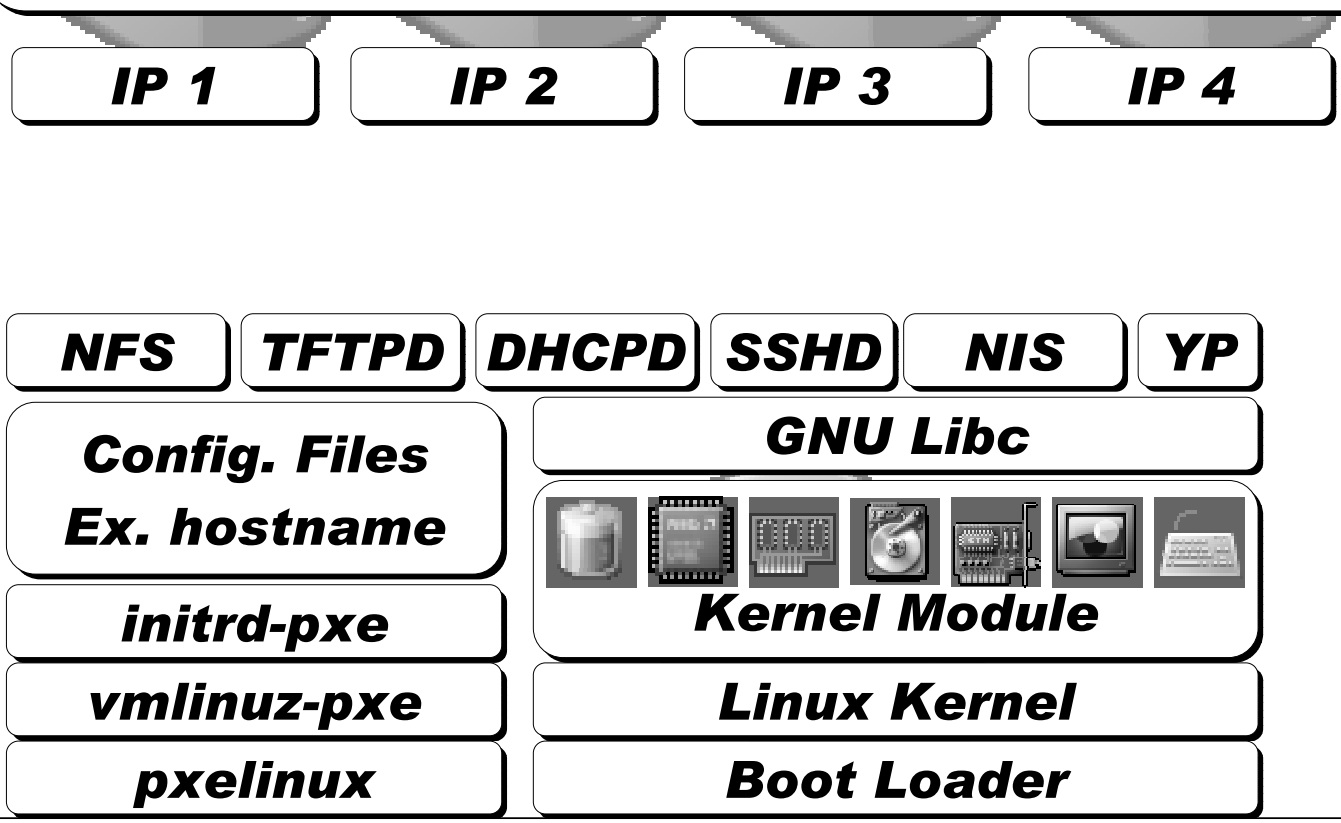
Linux Kernel

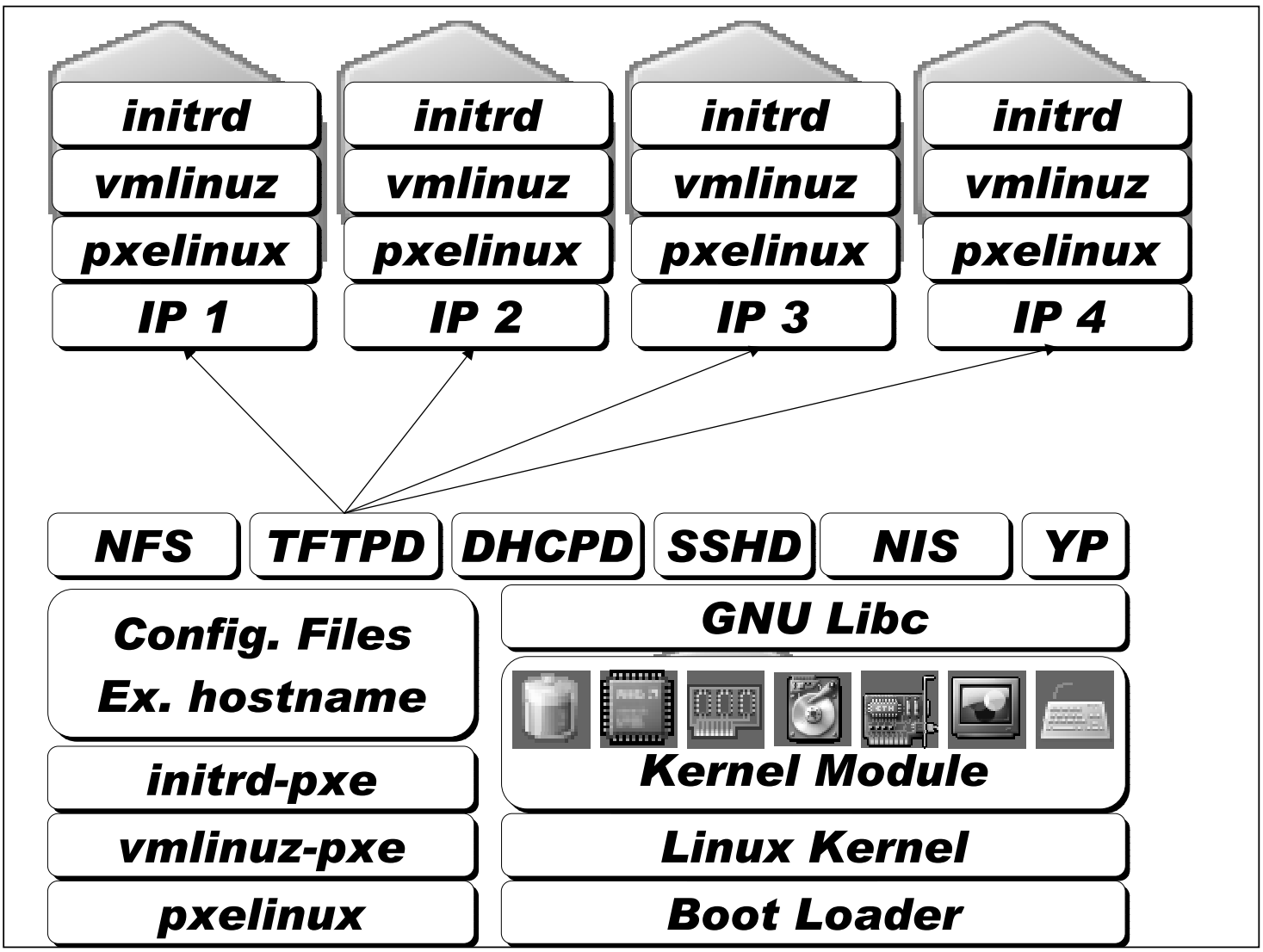
Boot Loader

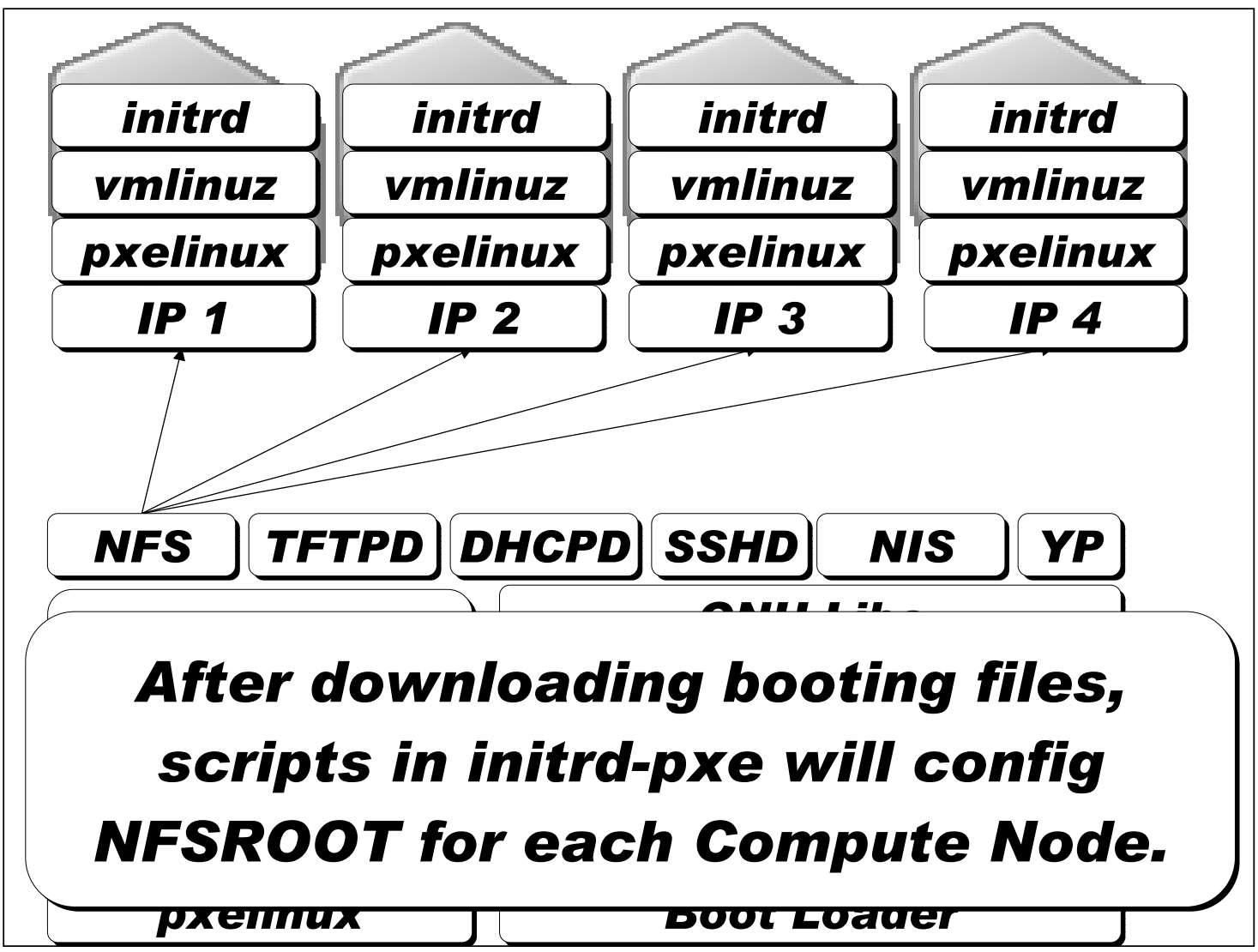
While Booting, PXE will query IP address from DHCPD.

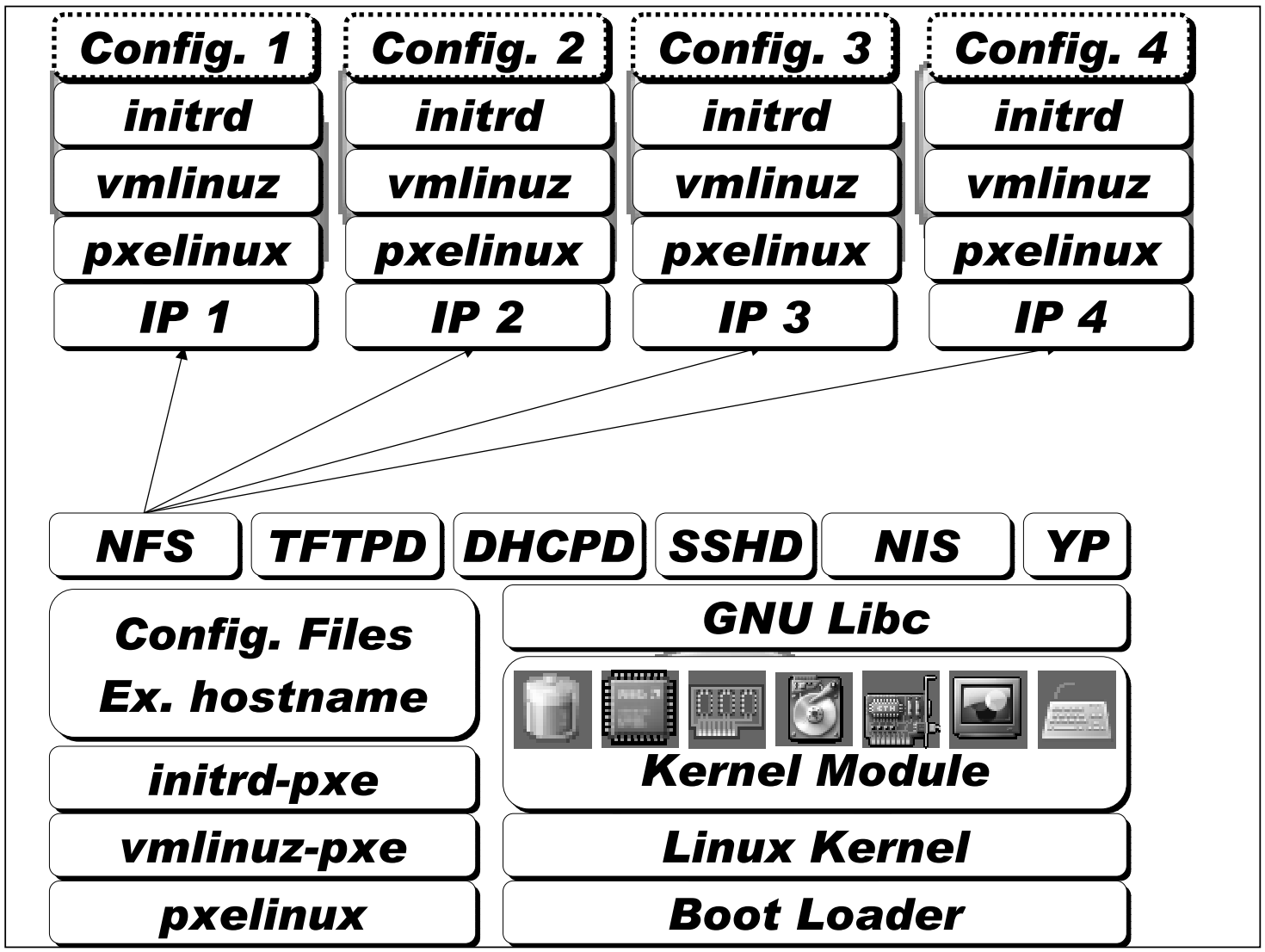


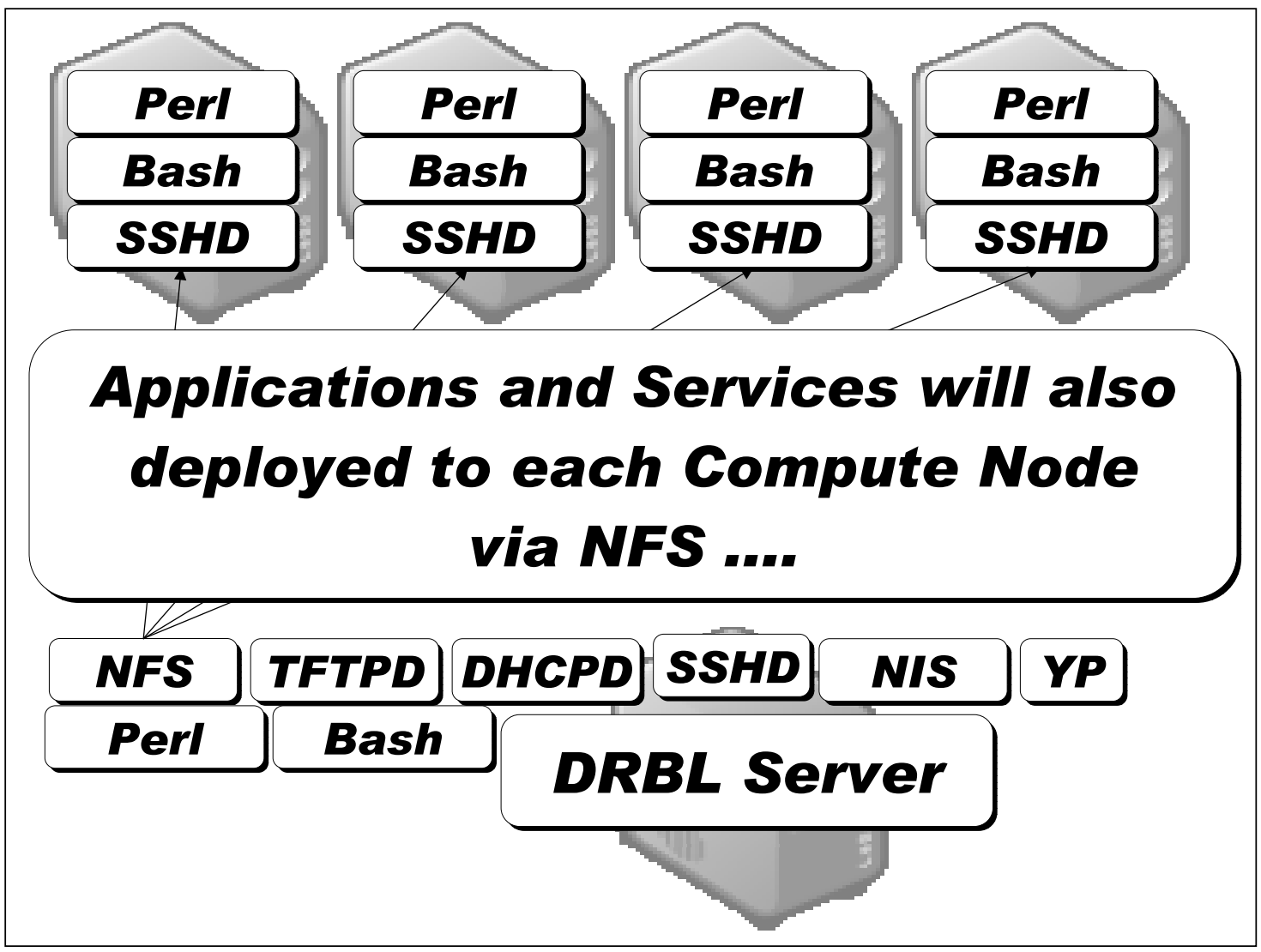
After PXE get its IP address, it will download booting files from TFTPD.

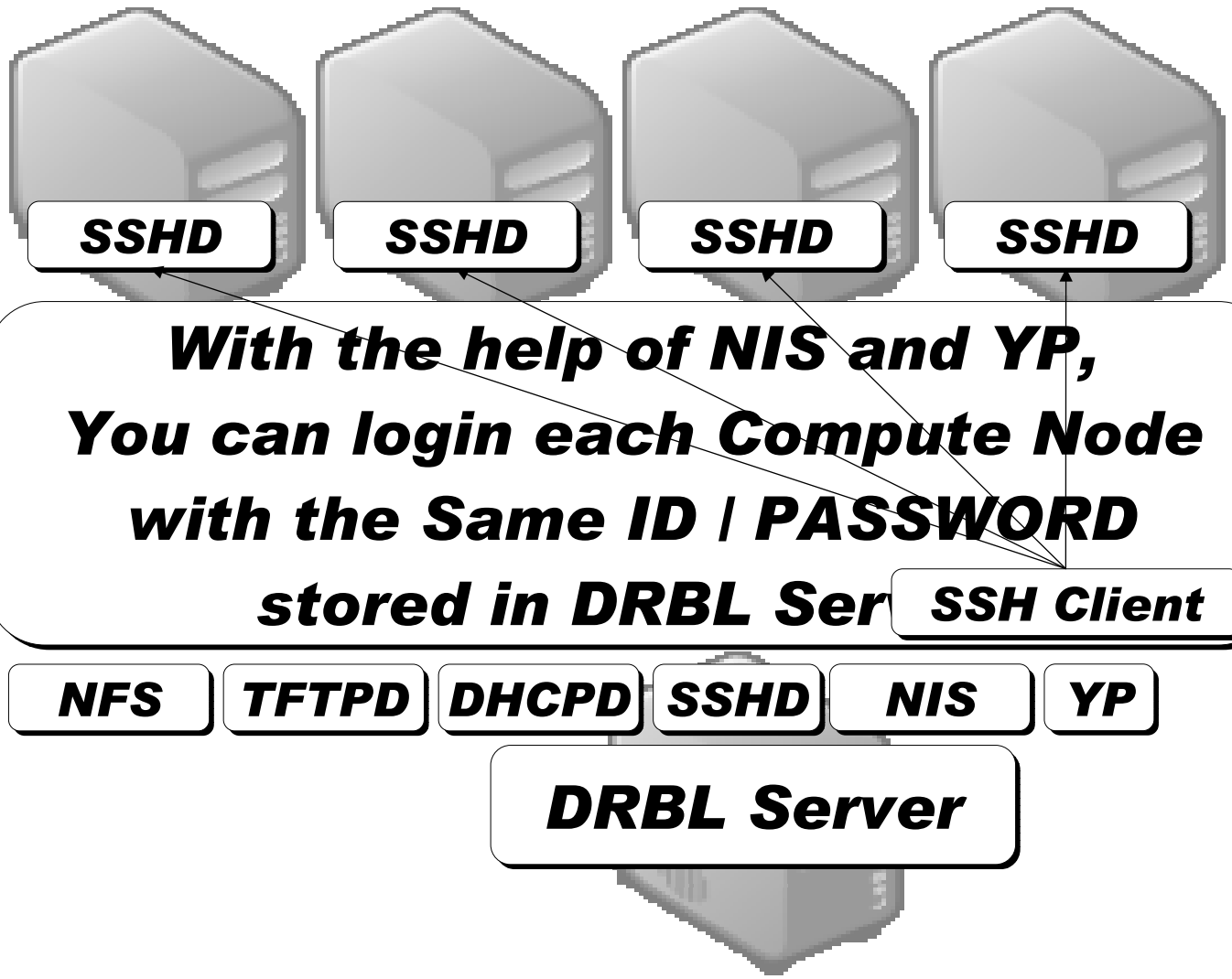


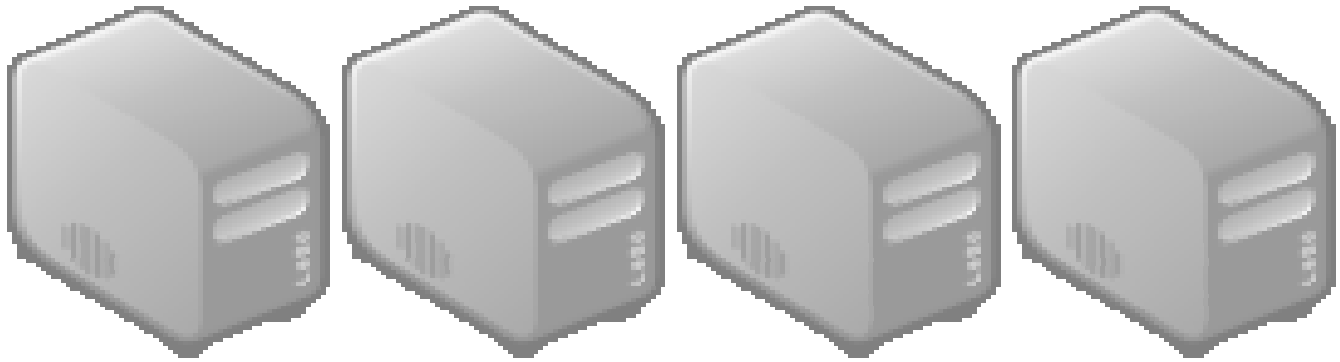

initrd
vmlinuz
pxelinux
IP 1
initrd
vmlinuz
pxelinux
IP 2
initrd
vmlinuz
pxelinux
IP 3
initrd
vmlinuz
pxelinux
IP 4
NFS
TFTPD
DHCPD
SSHD
NIS
YP
Config. Files
Ex. hostname
initrd-pxe
vmlinuz-pxe
pxelinux
GNU Libc
Kernel Module
Linux Kernel
Boot Loader


initrd
vmlinuz
pxelinux
IP 1
initrd
vmlinuz
pxelinux
IP 2
initrd
vmlinuz
pxelinux
IP 3
initrd
vmlinuz
pxelinux
IP 4
NFS
TFTPD
DHCPD
SSHD
NIS
YP
After downloading booting files, scripts in initrd-pxe will config NFSROOT for each Compute Node.


Config. 1
Config. 2
Config. 3
Config. 4
initrd
vmlinuz
pxelinux
IP 1
IP 2
IP 3
IP 4
NFS
TFTPD
DHCPD
SSHD
NIS
YP
Config. Files
Ex. hostname
initrd-pxe
vmlinuz-pxe
pxelinux
GNU Libc
Kernel Module
Linux Kernel
Boot Loader


Perl
Bash
SSHD
Applications and Services will also deployed to each Compute Node via NFS ....
NFS
TFTPD
DHCPD
SSHD
NIS
YP
Perl
Bash
DRBL Server





Questions?

Jazz Wang
Yao-Tsung Wang
jazz@nchc.org.tw

平行運算簡介／實例操作
企鵝也會的 **MPICH**

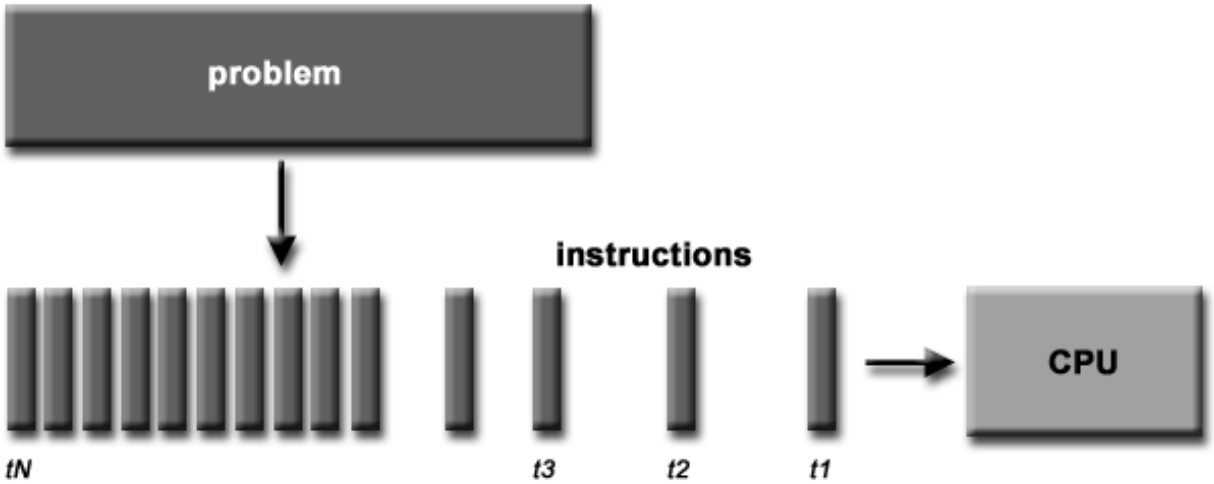
鄧偉華

wade@nchc.org.tw

財團法人國家實驗研究院
國家高速網路與計算中心
National Center for High-Performance Computing

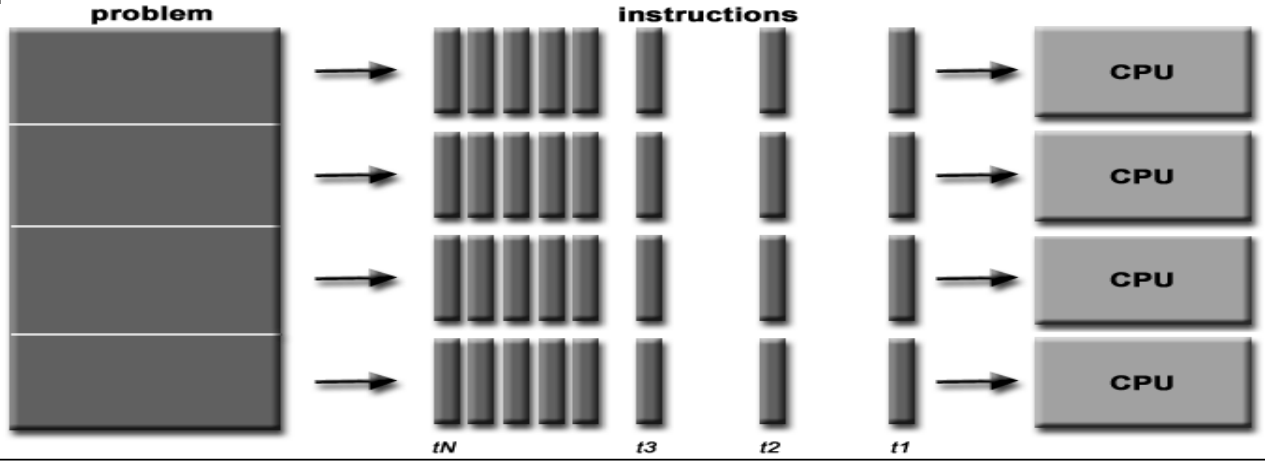
什麼是平行計算

- 傳統：
  - 單一程序
  - 單一 CPU



什麼是平行計算(續)

- 平行計算
  - 程序切割
  - 多 CPUs



爲什麼要平行計算

- 簡省時間
- 解決大型問題
- 即時性
- 使用更多來自網路上的資源
- 使用大量「便宜」 PCs 取代超級電腦
- 記憶體不足

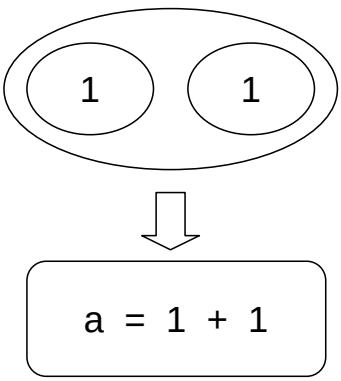
平行計算種類

- Flynn's taxonomy
- 多處理器架構

	單一程序 Single Instruction	多程序 Multiple Instruction
單一資料 Single Data	SISD	MISD
多資料 Multiple Data	SIMD	MIMD

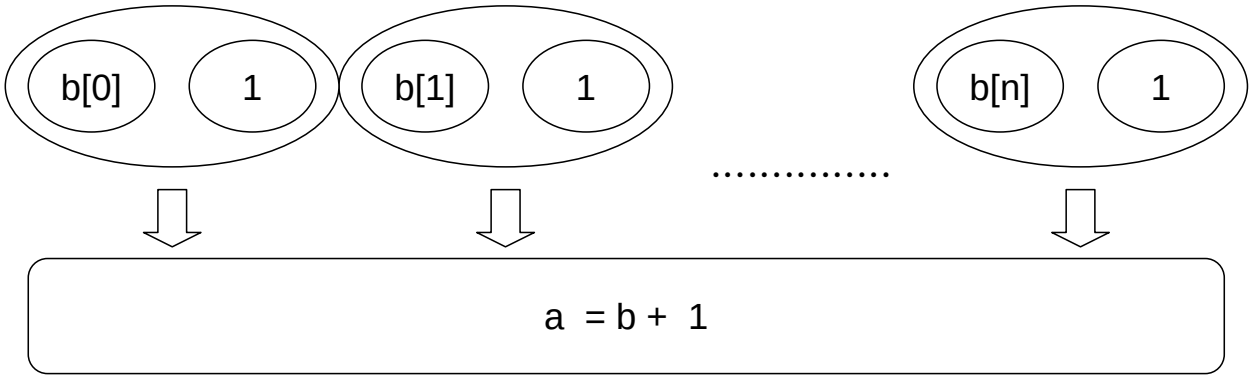
Single Instruction, Single Data (SISD)

- 非平行計算
- 單一程序
- 單一資料
- ex： a = 1 + 1



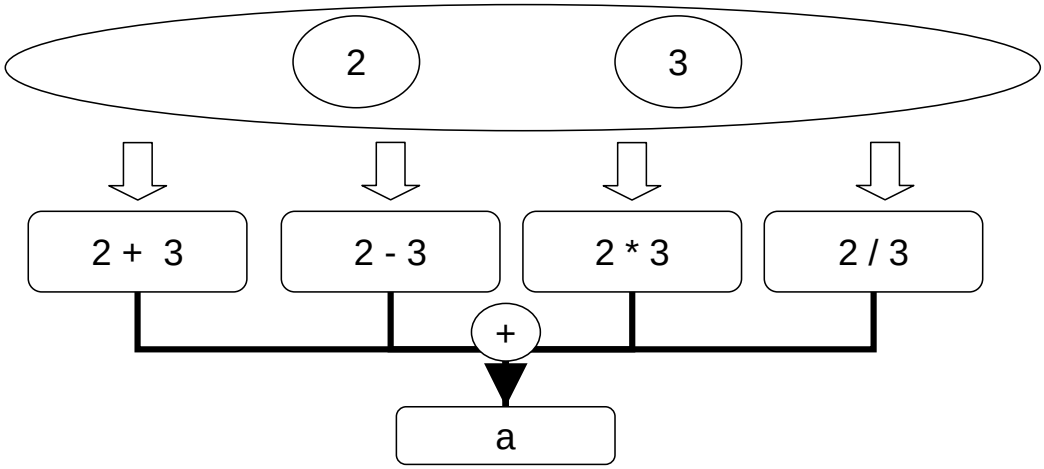
Single Instruction, Multiple Data(SIMD)

- 平行計算
- 單一程序
- 多資料
- ex： a = b + 1， a, b 是二組陣列，則同一時間就可以計算完成。而不需要算完 a[0] = b[0] + 1，再算 a[1] = b[1] + 1…再算 a[n] = b[n] + 1。



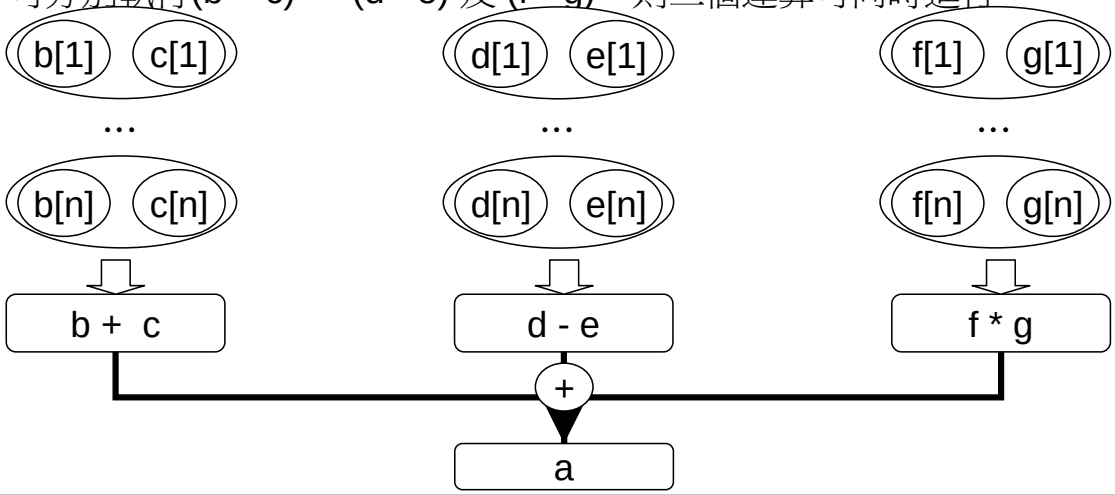
Single Instruction, Multiple Data(MISD)

- 平行計算
- 多程序
- 單一資料
- a = (2 + 3) + (2 - 3) + (2 * 3) + (2 / 3)



Multiple Instruction, Multiple Data(MIMD)

- 平行計算
- 多程序
- 多資料
- a = (b + c) + (d - e) + (f * g)， a, b, c, d, e, f, g 爲陣列如果有三臺機器可分別執行(b + c) 、 (d - e) 及 (f * g)，則三個運算可同時進行。

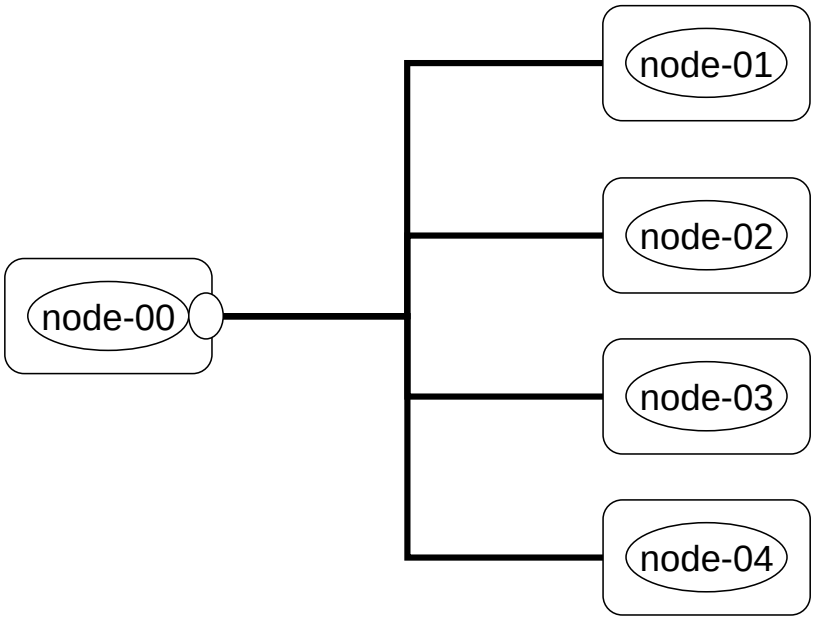


Point to point communication

- 同時間只有一個 node 傳訊息給另一個 node。

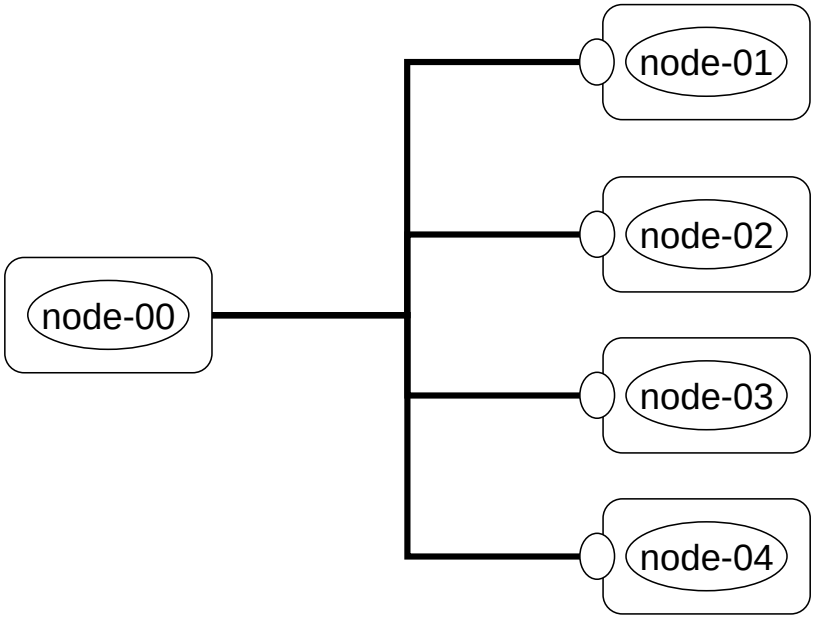
Point to point communication

- 可以有一個 node 丟訊息給多個 node



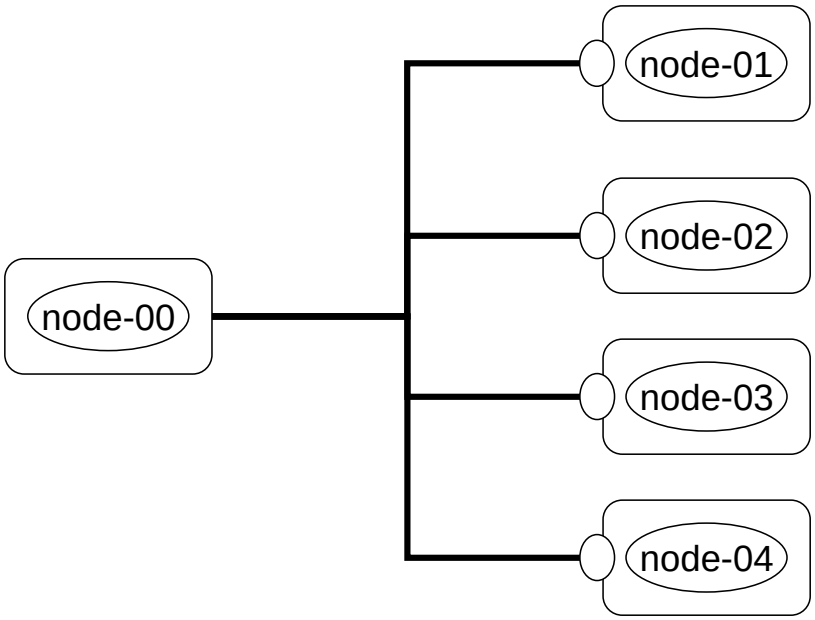
Point to point communication

- 也可以多個 node 丟訊息給一個 node



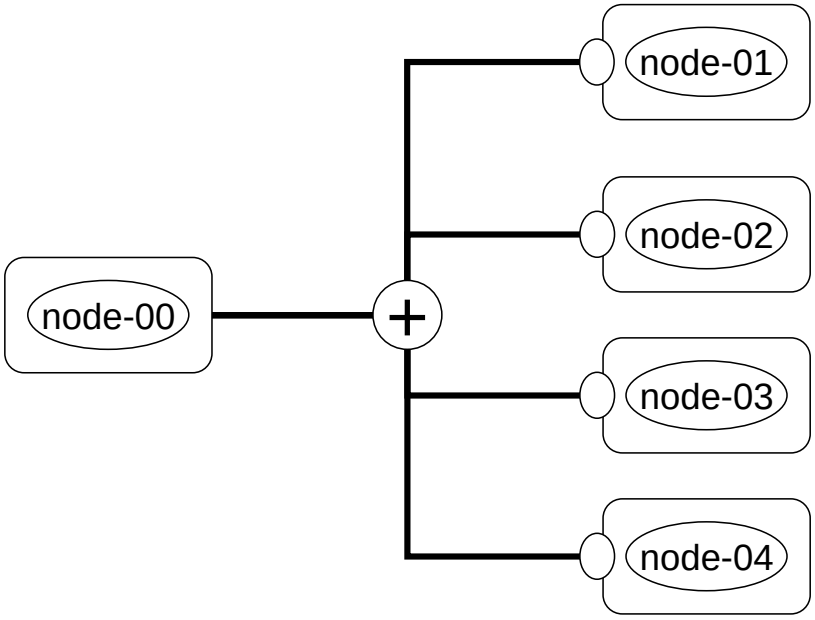
Collective communication

- 每個 node 同時傳訊息給 node 0



Collective communication

- 將每個訊息做完運算後再傳給 node 0



登入主機設定 **MPICH** 環境

- 登入主機。
  - ssh ymxx@140.129.162.12
- 輸入密碼。
- 產生 public key。
  - ssh-keygen -t rsa
  - 一直 enter.....enter.....
- 將 id_rsa.pub 更名爲 authorized_keys。
  - cp .ssh/id_rsa.pub .ssh/authorized_keys
- 設定 mpd.conf 。
  - echo "MPD_SECRETWORD=${user}$$" > ~/.mpd.conf

登入主機設定 **MPICH** 環境

- 更改 .mpd.conf 權限。
  - chmod 600 .mpd.conf
- 設定 mpd.hosts ，將 nodes IP 填入。
- for ((i=2;i<=7;i++)); do echo "192.168.129.$i" >> mpd.hosts; done
- 設定路徑
  - export PATH=$PATH:/opt/mpich2/bin
- 測試路徑是正確
  - which mpdboot
  - 正常顯示 /opt/mpich2/bin/mpdboot
- mkdir .dsh
- cp mpd.hosts .dsh/machines.list
- dsh -a hostname

啓動 **MPICH**

- 啓動 mpdboot ，並開啓 7 個 nodes
  - mpdboot -n 7
- 測試 nodes 是否正常啓動
  - mpdtrace
  - mpdringtest 1000
- 釋放 mpd
  - mpdallexit

如何執行 **MPICH** 平行程式

- 啓動 MPD
  - mpdboot -n 4 [-f machine_file]
  - -n how many mpds to start
  - -f hostsfile
- 列出所有 nodes
  - mpdtrace
- 執行
  - mpiexec -n 12 ./mpi/a.out
- 結束 MPD
  - mpdallexit

如何使用載入／啓動 MPICH

```
#include <stdio.h>
#include <mpi.h>

main (int argc, char **argv)
{
  MPI_Init(&argc, &argv);
  MPI_Comm_size (MPI_COMM_WORLD,  &numprocs);
  MPI_Comm_rank (MPI_COMM_WORLD,  &myid);
    . . .
    . . .
  MPI_Finalize();
  return 0;
}
```

啟動該程式在多個CPU上的平行計算工作

得知參與平行計算的CPU個數 (nproc)

得知我是第幾個CPU (myid) from 0

結束平行計算工作

HELLO MPICH

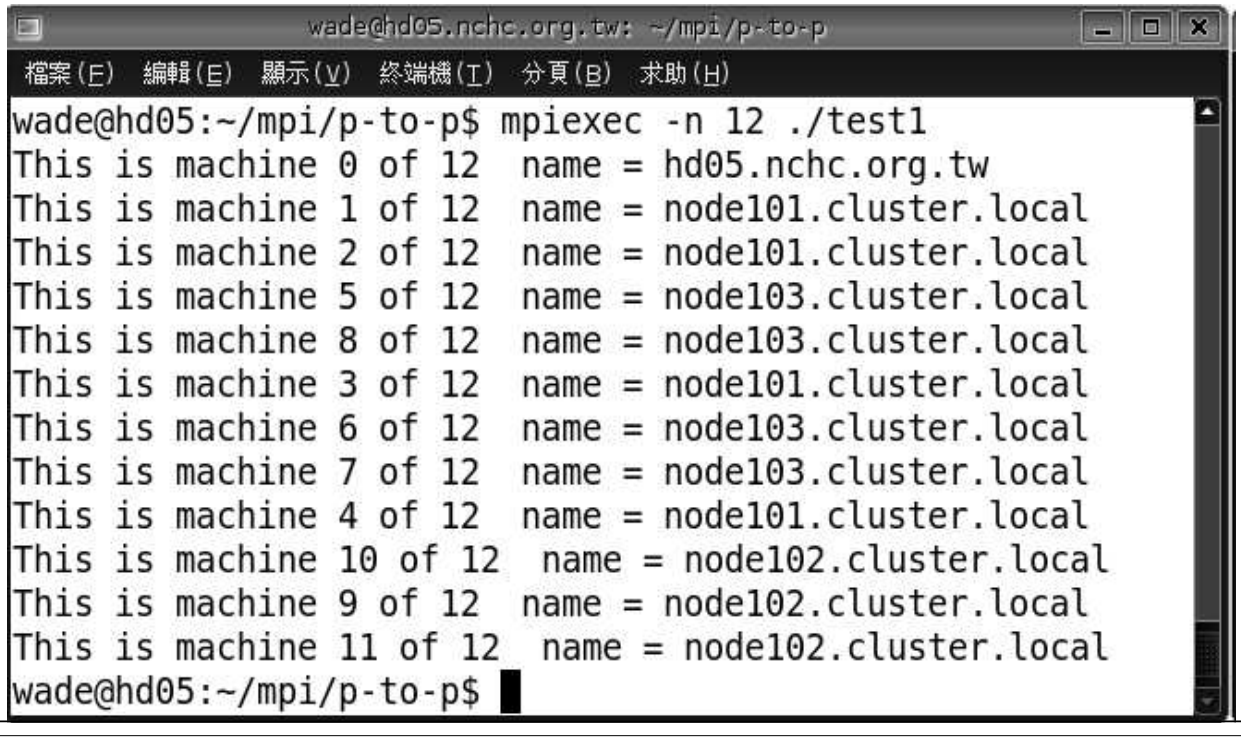
目標：

- 每個 node 將自己的 id 印出，並且將所有的參與活動的 node 總數也印出，顯示出自己的主機名稱。

HELLO MPICH

```
#include <stdio.h>
#include <mpi.h>
main (int argc, char **argv) {
 int rank, size, len;
 char name[MPI_MAX_PROCESSOR_NAME];
 MPI_Init(&argc, &argv);
 int myid, numprocs;

 /* 取得 node 總數 */
 MPI_Comm_size(MPI_COMM_WORLD,&numprocs);
 /* 取得本身 node id / rank */
 MPI_Comm_rank(MPI_COMM_WORLD,&myid);
 /* 取得本身 host name */
 MPI_Get_processor_name(name, &len);
 printf("This is machine %d of %d name = %s\n“, myid, numprocs, name);

 MPI_Finalize();
 }
}
```

結果



MPI_Send

MPI_Send ((void *)&data, icount, DATA_TYPE, idest, itag, MPI_COMM_WORLD)

- data 要送出去的資料起點，可以是純量 (scalar) 或陣列 (array) 資料。
- icount 要送出去的資料數量，當icount的值大於一時，data必須是陣列。
- DATA_TYPE 要送出去的資料類別，MPI 內定的資料類別。
- idest 是收受資料的CPU id。
- itag 要送出去的資料標籤。
- MPI_COMM_WORLD 通信域。

MPI_Recv

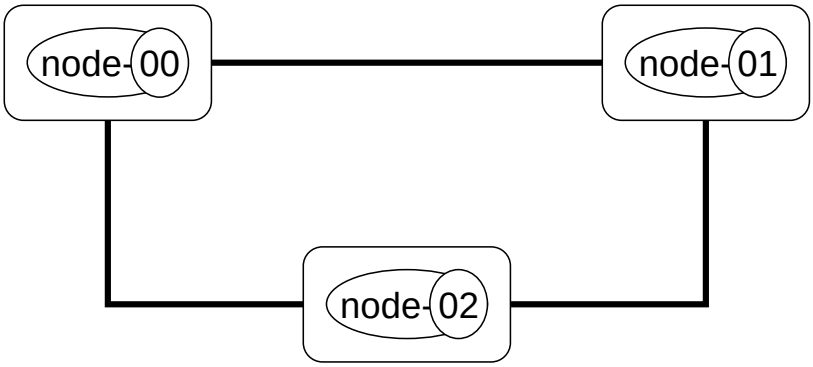
MPI_Recv(void *buf, int count, MPI_Datatype datatype, int source, int tag, MPI_Comm comm, MPI_Status *status)

- buf 要接收的資料起點，可以是純量 (scalar) 或陣列 (array) 資料。
- icount 要接收的資料數量，當icount的值大於一時，data 必須是陣列。
- DATA_TYPE 要接收的資料類別，MPI 內定的資料類別。
- source 是收受資料的 CPU id。
- itag 要接收的資料標籤。
- MPI_COMM_WORLD 通信域。

範例

目標：

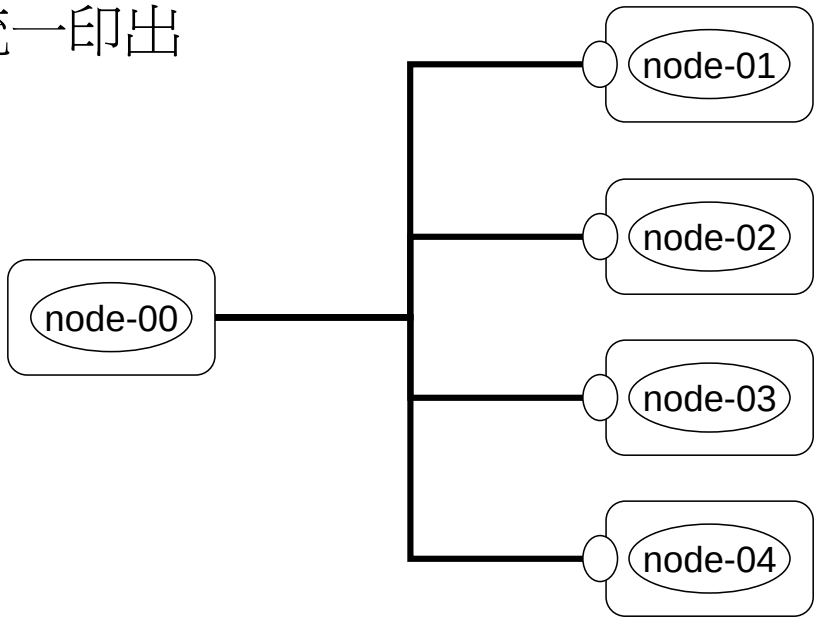
- 每個 node 將自己的 id 資訊送給下一個 node。




```
#include <mpi.h>
#include <stdio.h>
main(int argc,char **argv) {
  int n, myrank, numprocs;
  MPI_Status status;
  MPI_Init(&argc,&argv);
  MPI_Comm_size(MPI_COMM_WORLD,&numprocs);
  MPI_Comm_rank(MPI_COMM_WORLD,&myrank);

  /* node 0 will send the first message */
  if(myrank == 0) {
    n = myrank;
    MPI_Send(&n, 1, MPI_INT, 1, 99, MPI_COMM_WORLD);
    printf("[Ndde %d]「%d」 >> [Node %d]\n\n", myrank, n, myrank+1);
  }

  /* node 1 to node n-2 will send message to the next node */
  if(myrank>0 && myrank<numprocs-1) {
    MPI_Recv(&n, 1, MPI_INT, myrank-1, 99, MPI_COMM_WORLD, &status);
    printf("[Node %d] << 「%d」[Node %d]\n", myrank, n, status.MPI_SOURCE);
    n = myrank; MPI_Send(&n, 1, MPI_INT, myrank+1, 99, MPI_COMM_WORLD);
    printf("[Ndde %d]「%d」 >> [Node %d]\n\n", myrank, n, myrank+1);
  }

 /* the final node n-1 will not send any message but receive*/
  if(myrank==numprocs-1) {
    MPI_Recv(&n, 1, MPI_INT, myrank-1, 99, MPI_COMM_WORLD, &status);
    printf("[Node %d] << 「%d」[Node %d]\n", myrank, n, status.MPI_SOURCE);
    }

  MPI_Finalize();
}
```

動手作

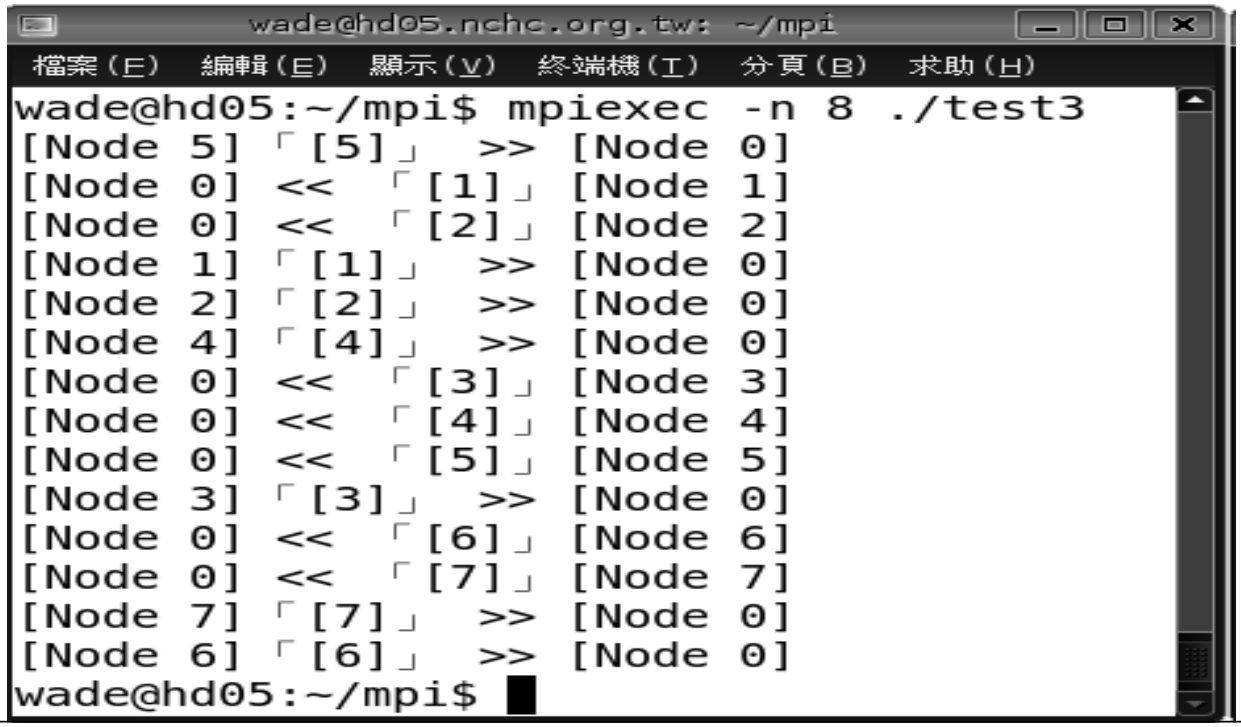
- 每個 node 將訊息傳送給 node 0，由，node 0 統一印出



提示

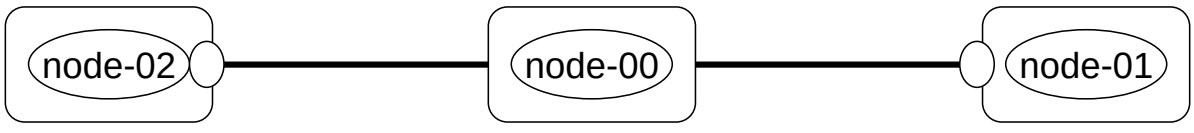
```
main(int argc, char **argv){
  /* Node 0 will do the following */
  if(myrank == 0) {
    /* receive messages from other nodes */
    ……
    MPI_Recv();
  }

  /* other Nodes will do the following */
  if(myrank != 0) {
    /* send node's rank to Node 0 */
    MPI_Send(message, 20, MPI_CHAR, 0, 99, MPI_COMM_WORLD);
  }
```

結果



再動手作

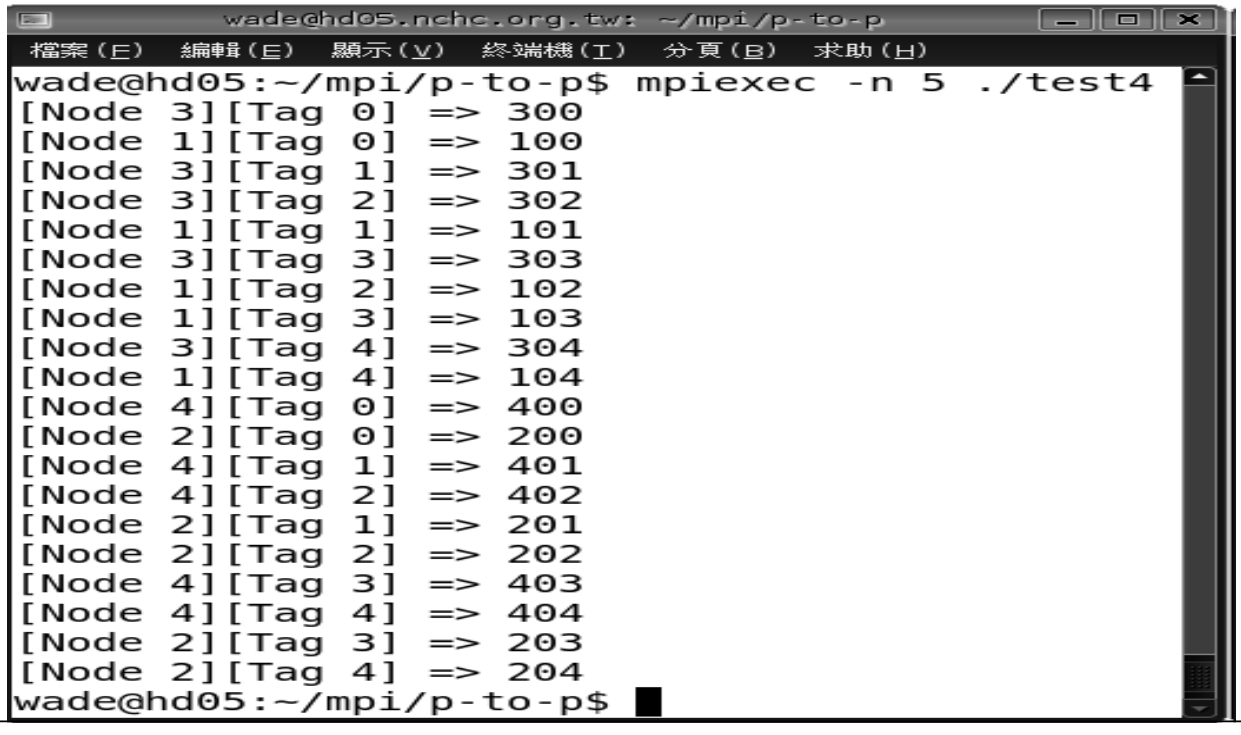
- 讓 node 0 可以接受來自任何 node 的訊息，每個 node 將訊息標上不同 tag (0, 1, 2...)後傳給 node 0



提示

- MPI_ANY_SOURCE 接收來自任何 node 訊息。
- MPI_ANY_TAG 接收任何 tag。

結果



MPI_Reduce

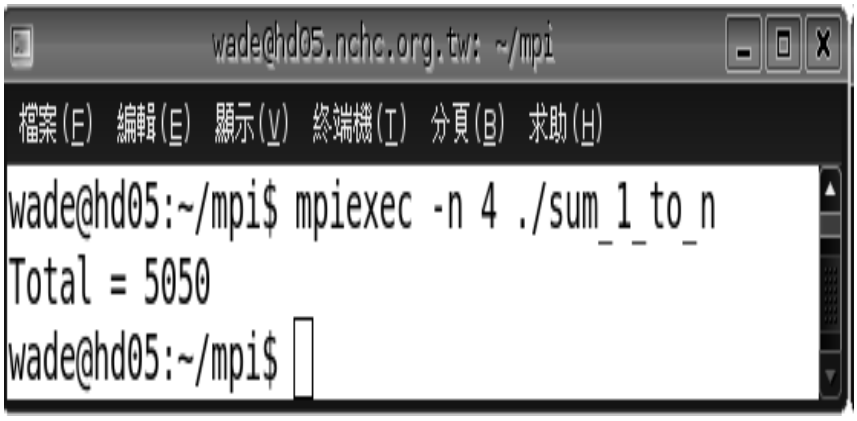
MPI_Reduce (void *sendbuf, void *recvbuf, int count, MPI_Datatype datatype, MPI_Op op, int root, MPI_Comm comm)

- sendbuf address of send buffer (choice)
- count number of elements in send buffer (integer)
- datatype data type of elements of send buffer (handle)
- op reduce operation (handle)
- root rank of root process (integer)
- comm communicator (handle)

計算 1 + 2 + 3 + ... + 100

```
#include <stdio.h>
#include <mpi.h>
main (int argc, char **argv) {
  int rank, size, i; int myid, numprocs; int myTotal = 0;
  int Total = 0; MPI_Init(&argc, &argv);
  MPI_Comm_size(MPI_COMM_WORLD, &numprocs);
  MPI_Comm_rank(MPI_COMM_WORLD, &myid);
  for(i = 1; i <= 100; i++) {
    if((i % numprocs) == myid) {
      myTotal += i;
    }
  MPI_Reduce(&myTotal, &Total, 1, MPI_INT, MPI_SUM, 0,MPI_COMM_WORLD);
  }
  if(myid==0) {
    printf("Total = %d\n", Total);
  }
  MPI_Finalize();
}
```

結果



動手作

- 計算 **Fibonacci number** 從第 **1** 項加至第 **n** 項總合。
- **Fibonacci number**
  - **f(n) = f(n - 1) + f(n - 2)**
  - **f(0) = 1, f(1) = 1**
  - **1, 1, 2, 3, 5, 8, ...**

MPI_Bcast

MPI_Bcast (void *buffer, int count, MPI_Datatype datatype, int root, MPI_Comm comm)

- buffer 要傳送的資料起點，可以是純量 (scalar) 或陣列 (array) 資料。
- count 要傳送的資料數量，當icount的值大於一時，data 必須是陣列。
- datatype 要傳送的 (buffer) 資料類別，MPI 內定的資料類別。
- root root 的 id (rank) 是多少
- comm 通信域

MPI_Barrier

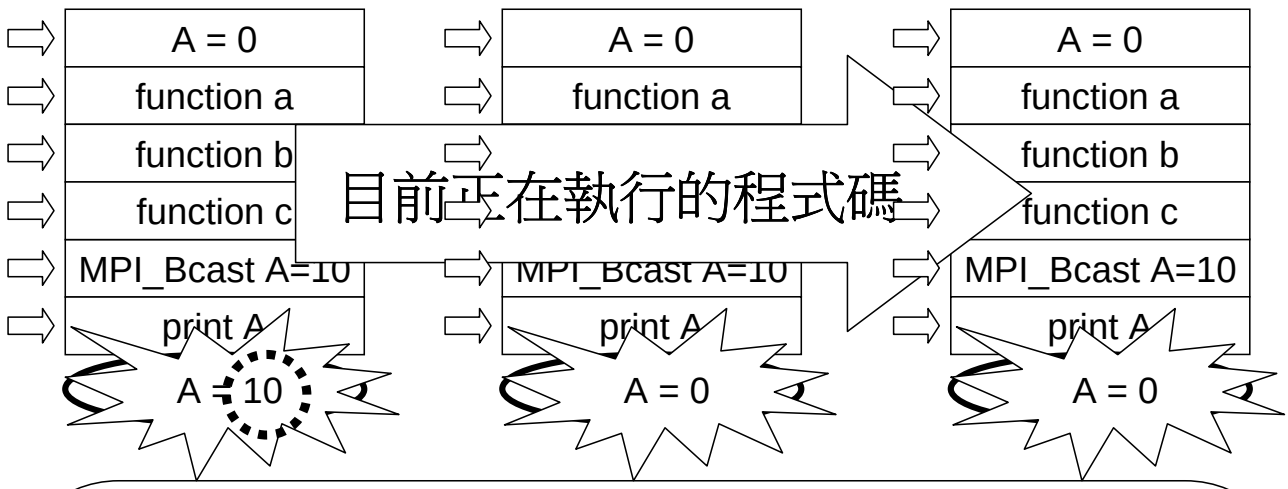
MPI_Barrier (MPI_Comm comm)

- comm 通信域

爲什麼需要 **MPI_Barrier**

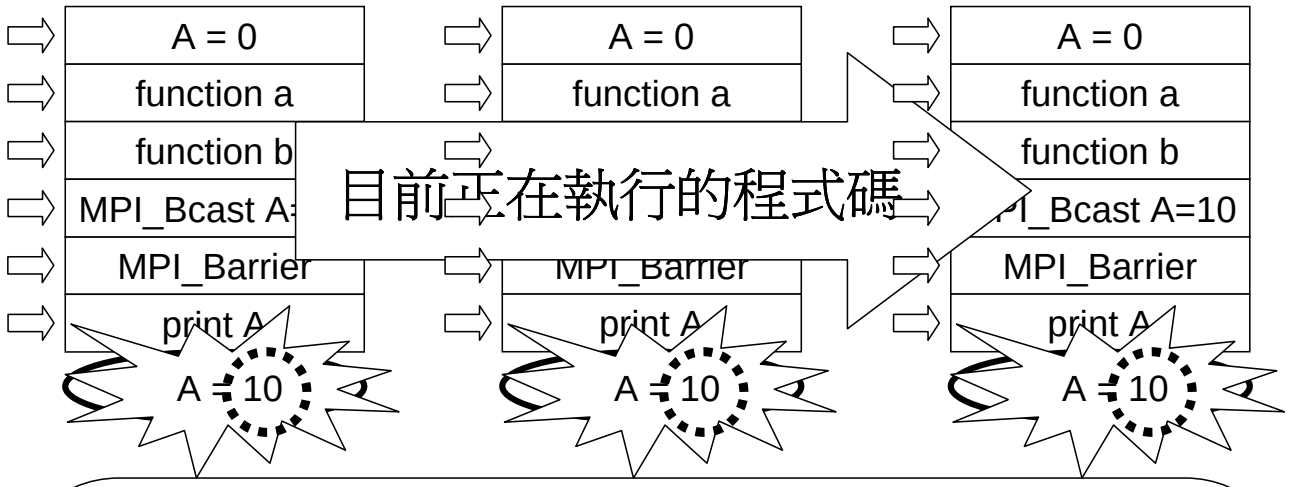
- MPI_Barrier 是爲了讓所有 nodes 同步
- 只有當所有 nodes 都執行完 MPI_Barrier 時，所以的 nodes 才再繼續往下執行下面的程式碼。

無 **MPI_Barrier**



node 0 爲 root ，負責將 A = 10 廣播出去，則有可能出現以下情形：
1、node 0 順利比其它 nodes 先執行到 MPI_Bcast A=10，將 A = 10 送出
2、node 0 還未執行 MPI_Bcast A=10 ，其它 nodes 已經執行 print A，
則其它 nodes 中 A 的結果就不會同步爲 10。

有 **MPI_Barrier**

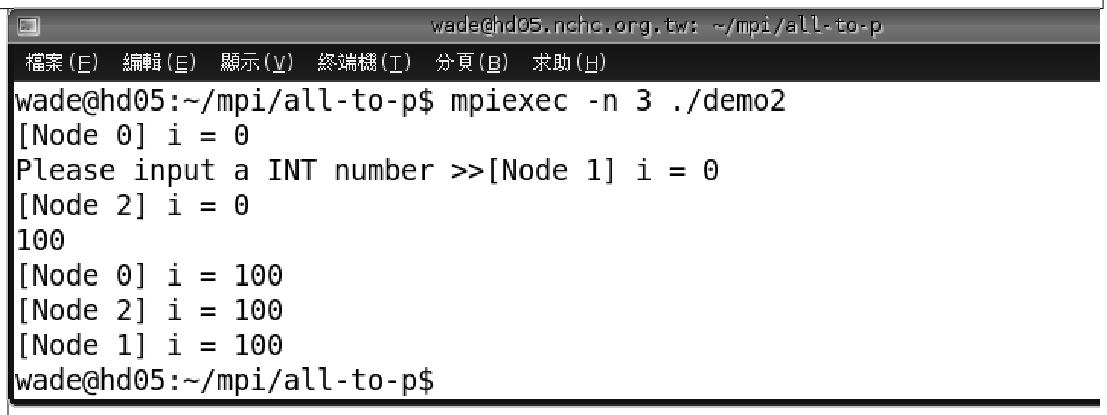


此時不管那個 nodes ，皆會在 MPI_Barrier 等待，直至全部 nodes 執行完 MPI_Barrier 時才會繼續往下執行程式碼，如此一來才可確保每個 root 已經將 A = 10 廣播出去，使每個 nodes 的 A 都改爲10。

範例

```
#include <stdio.h>
#include <mpi.h>
main (int argc, char **argv) {
  int i = 0;
  int myid, numprocs;
  MPI_Init(&argc, &argv);
  MPI_Comm_size(MPI_COMM_WORLD, &numprocs);
  MPI_Comm_rank(MPI_COMM_WORLD, &myid);
  /* 這邊先印出所有 nodes 的整數 i 初始狀態 */
  printf("[Node %d] i = %d \n", myid, i);
  /* node 0 (root) 會要求使用者輸入一個 INT 的整數 */
  if (myid == 0) {
    printf("Please input a INT number >>");
    scanf("%d", &i);
   }
  /* node 0 (root) 會將所有 nodes 的 i 的值設爲使用者所輸入的數 */
  MPI_Bcast(&i, 1, MPI_INT, 0, MPI_COMM_WORLD);
  /* 所有的 nodes 在此同步，以確認 node 0 (root) 已將 i 送給所有 nodes */
  MPI_Barrier(MPI_COMM_WORLD);
  printf("[Node %d] i = %d \n", myid, i);
  MPI_Finalize();
}
```

結果



我們可以發現到第二行很明顯的不同，這就是前面介紹，不一定誰會先執行到 PRINT 的指令，所以我在第四行才能輸入 100 然後底下每個 node 的 i 都被設爲 100。

MPICH 設計注意事項

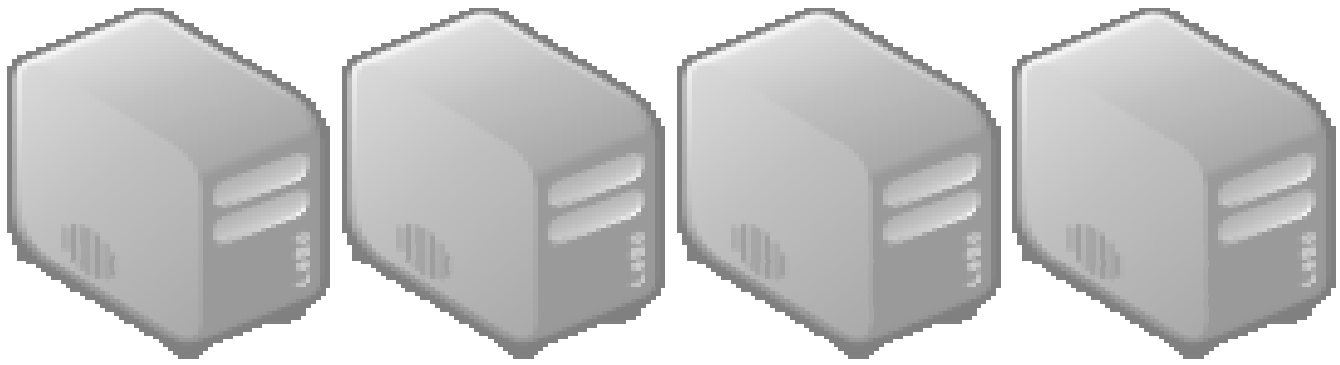
- MPICH 屬於 SIMD 架構
- 指令及資料量切割
  - 優：有助於平行運算
  - 缺：增加傳輸次數及傳輸量

Q & A

- website：
  - http://trac.nchc.org.tw/grid/wiki/mpich
- mail：
  - wade@nchc.org.tw

參考

- http://www-unix.mcs.anl.gov/mpi/mpich1/docs.html
- http://www-unix.mcs.anl.gov/mpi/www/www3/
- http://www.mpi-forum.org/



第二天課程重點回顧

Jazz Wang
Yao-Tsung Wang
jazz@nchc.org.tw

What we learn today ?

Q1: What is HPC? 何謂高速運算？
A1: HPC就是結合電腦硬體、軟體和一堆專家，
用各種方式解決困難的問題。

Q2: Types of HPC? 高速運算的種類有哪些？
A2: Mainframe, PC Cluster, Parallel, Distributed, Grid, Cloud
超級電腦、電腦叢集、平行、分散、格網、雲端運算

Q3: Can HPC solve all your problems? 高速運算可以解決所有問題?
A3: No. 高速運算無法解決所有問題，各種類別也各有所長。

Q4: What is PC Cluster? 何謂電腦叢集?
A4: Cluster = lots of PCs. 很多電腦用內部網路串起來，就是叢集。

Q5: What is Grid? 何謂格網運算?
A5: Grid = Cluster of Cluster. 把好幾座叢集視為一座抽象的叢集。

Q6: What is Cloud? 何謂雲端運算?
A6: Cloud = lots of Virtual Cluster. 在實體叢集中打造多座虛擬叢集。

階段目標

- 學習何謂電腦叢集運算與openPBS排程實作
- 學習如何用再生龍進行系統備份
- 學習何謂平行運算
- 學習如何使用MPICH 撰寫平行程式