Dec 7, 2010

CRAWLZILLA

雲端運算架構 Hadoop 與其搜尋引擎之應用

陳威宇、楊順發、郭文傑

crawlzilla@ NCHC

TAIWAN

www.nchc.org.tw

National Applied Research Laboratories

NCHC

什麼是雲端運算啊？
What is Cloud Computing ?

http://www.youtube.com/watch?v=Z5f2FQkLfdo (中文版)
http://www.youtube.com/watch?v=ae_DKNwK_ms (原文版)

The wisdom of Clouds (Crowds)

雲端序曲：雲端的智慧始終來自於群眾的智慧

2006年8月9日

Google執行長施密特（Eric Schmidt）於SES'06會議中首次使用「雲端運算（Cloud Computing）」來形容無所不在的網路服務

2006年8月24日

Amazon以Elastic Compute Cloud命名其虛擬運算資源服務

http://www.cnet.co.uk/i/c/blg/cat/software/cloudcomputing/clouds1.jpg

One key spirit of Cloud Computing
用一句話說明雲端運算!服務才是王道!

Anytime 隨時

Anywhere 隨地

Key spirit of Cloud ~
形成服務才是重點!!
Everything as a Service !!

With Any Devices 使用任何裝置

Accessing Services 存取各種服務

Cloud Computing =~ Network Computing

雲端運算 =~ 網路運算

2 R&D directions : Cloud or Device
兩大研究方向：你該選「雲」還是「端」？

端

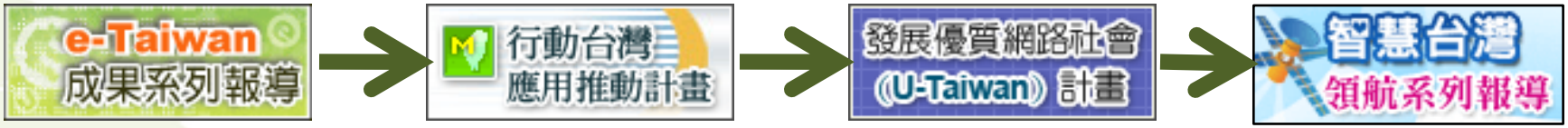
symbian
OS

集中，大廠
Centerized，
Enterprise

多元，中小廠
Diversify，
SMB

Evolution of Cloud Services

雲端服務 -> 軟體演化史的趨勢



實體 Physical	單機版 個人使用 Personal Software	網路版 多人共享 Share Service Software	行動版 隨時存取 Mobile Cloud Service
信箱 Mailbox	E-Mail	Web Mail	Mobile Mail
電視 TV	電視盒 Setop Box	Web TV Ex. Youtube	Mobile TV
打字機 Typer Writer	Office	Google Docs	M-Office
電話 Telephone	數位電話 PBX	Skype	Flash Wengo
佈告欄 Bullet Borad	電子佈告欄 BBS	部落格 Blog	微網誌 Twitter

Cloud Computing =~ Network Computing ！

雲端運算 =~ 網路運算 ！

Network Computing =~ Parallel And Distributed Computing？

網路運算 =~ 平行分散式運算？

Parallel And Distributed Computing =~ Cloud Computing ？？

平行分散式運算 =~ 雲端運算 ？？

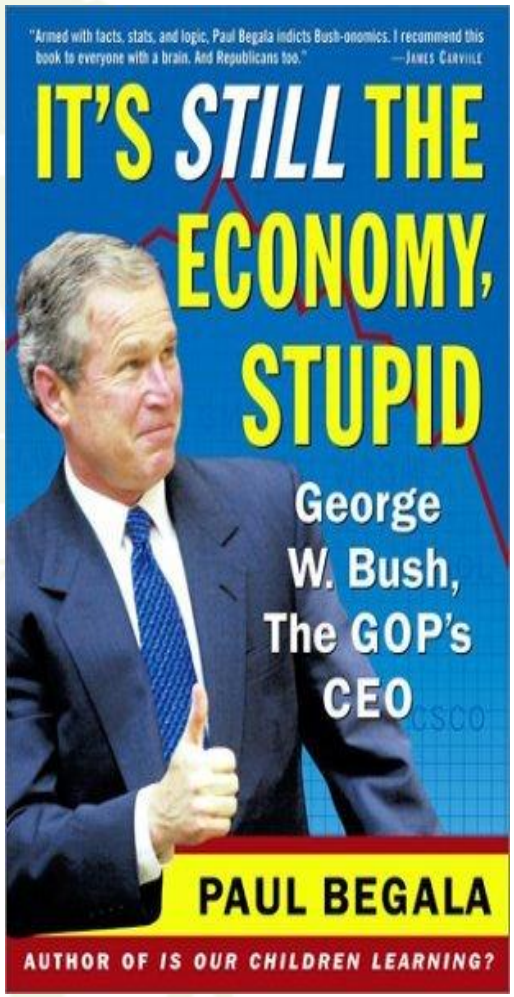
Computing with big datasets

is a fundamentally different challenge than doing "big compute" over a small dataset

Parallel And Distributed Computing
平行分散式運算

- **Grid computing (格網運算)**
  - MPI, PVM, Condor…
  - 著重於: 分散工作量
- **目前的問題在於：如何分散運算大資料量**
  - 試想用分散在世界各地的grid，運算100GB的資料…
  - 交換資料需同步處理 => Dead-Lock !
  - 有限的頻寬

IT'S THE DATA, STUPID!



「笨蛋！重點在經濟」

（**"It's the economy, stupid"**）

這句標語促使柯林頓當上美國第**42**屆總統。

- 1992 年

「笨蛋！重點還是在經濟」

（**"It's STILL the economy, stupid"**）

- 2002 年

雲端時代，谷歌會說：

"It's the data, stupid"

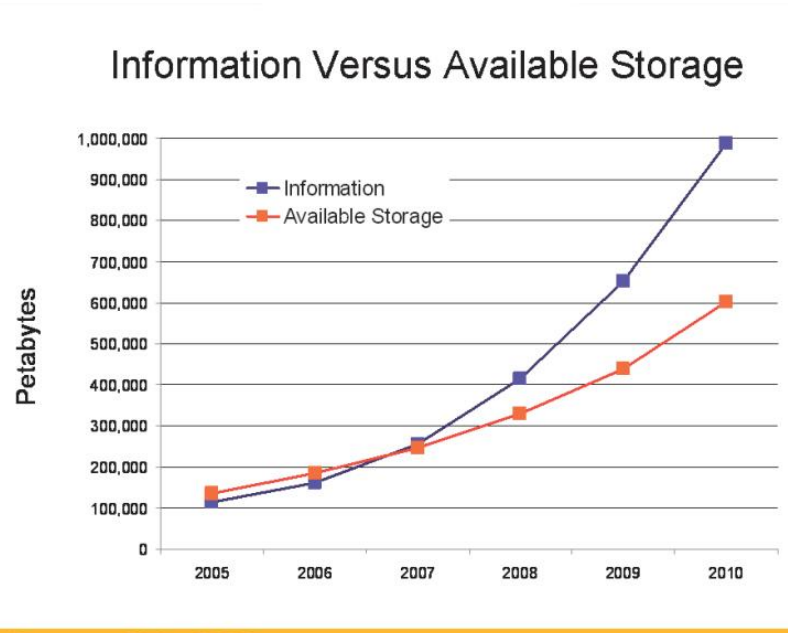
誰掌握了你的資料，就有機會掌握你的荷包

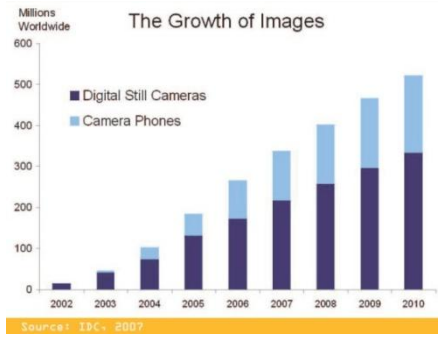
電腦、手機掉了，您心疼的是甚麼呢？

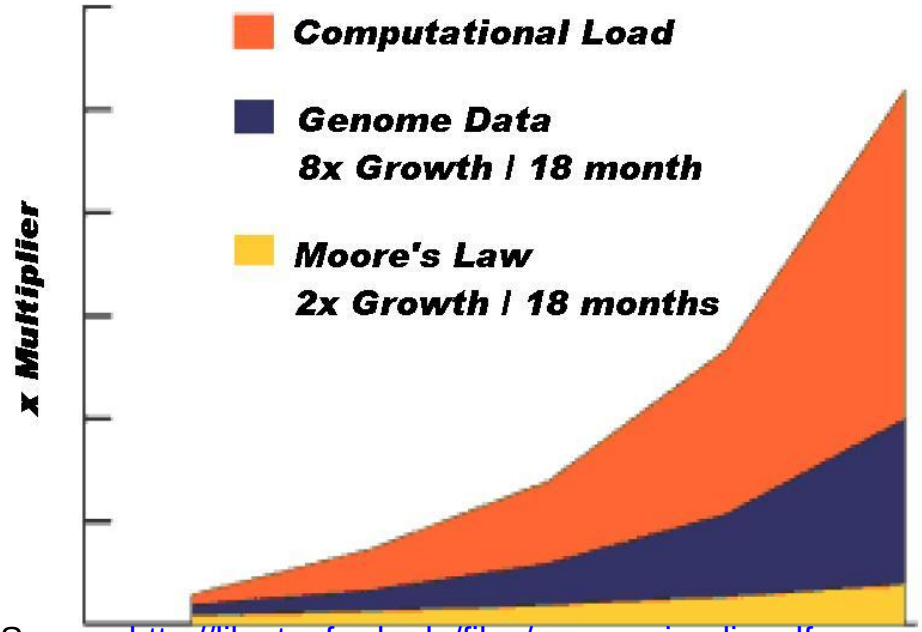
- 2007 年

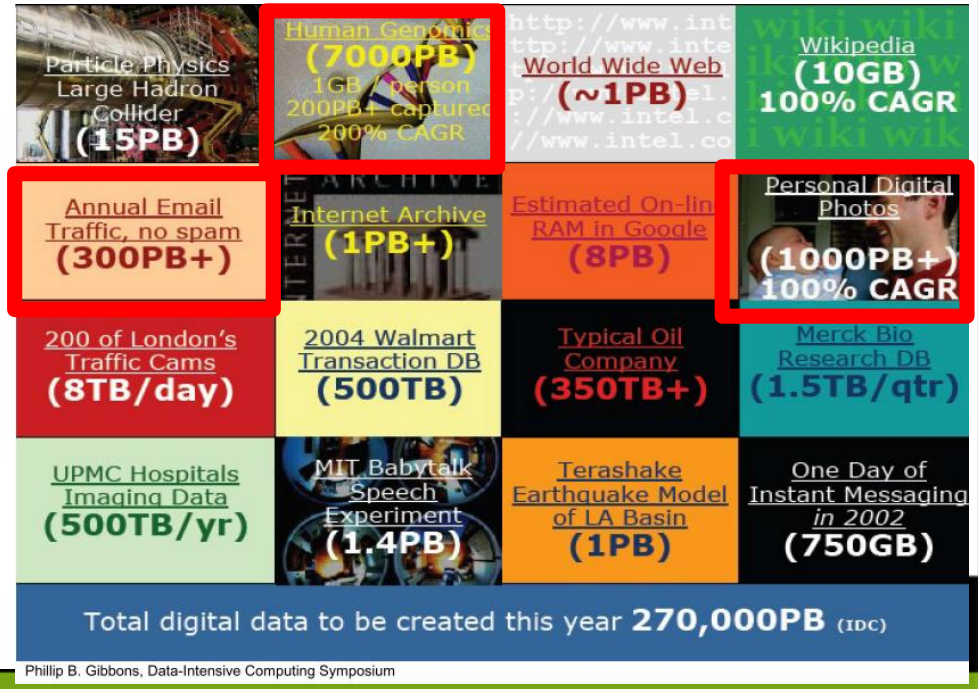
2007 Data Explore

Top 1 : Human Genomics – 7000 PB / Year
Top 2 : Digital Photos – 1000 PB+/ Year
Top 3 : E-mail (no Spam) – 300 PB+ / Year









Source: http://lib.stanford.edu/files/see_pasig_dic.pdf

Big Data Set Bottle Neck
大量的資料瓶頸

- **Data processed by Google every month: 400 PB … in 2007**
  - Max data in memory: 32 GB
  - Max data per computer: 12 TB
  - Average job size: 180 GB
- **光一個device的讀取時間= 45 minutes**

Big Data Parallelizing
平行大量的資料

- 運算資料可以很快速，但瓶頸在於硬碟的 **I/O**
  - 1 HDD = 75 MB/sec
- 解法: **parallel reads**
  - 1000 HDDs = 75 GB/sec

Three Core Technologies of Google ....
Google的三大關鍵技術 ....

- Google在一些會議分享他們的三大關鍵技術
- **Google shared their design of web-search engine**
  - SOSP 2003 :
  - "The Google File System"
  - http://labs.google.com/papers/gfs.html
  - OSDI 2004 :
  - "MapReduce : Simplifed Data Processing on Large Cluster"
  - http://labs.google.com/papers/mapreduce.html
  - OSDI 2006 :
  - "Bigtable: A Distributed Storage System for Structured Data"
  - http://labs.google.com/papers/bigtable-osdi06.pdf

Open Source vs Google Technologies

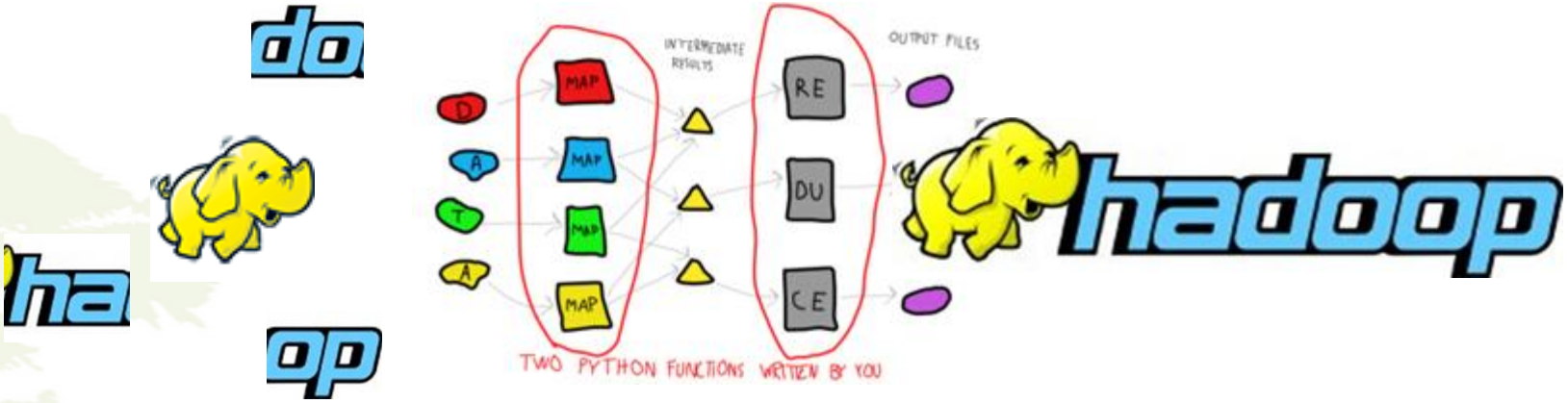
Google三大關鍵技術對應的自由軟體

BigTable A huge key-value datastore	HBase, Hypertable Cassandra, ...
MapReduce To parallel process data	Hadoop MapReduce API Sphere MapReduce API, ...
Google File System To store petabytes of data	Hadoop Distributed File System (HDFS) Sector Distributed File System

更多不同語言的MapReduce API實作：
http://trac.nchc.org.tw/grid/intertrac/wiki%3Ajazz/09-04-14%23MapReduce

其他值得觀察的分散式檔案系統：

- IBM GPFS - http://www-03.ibm.com/systems/software/gpfs/
- Lustre - http://www.lustre.org/
- Ceph - http://ceph.newdream.net/



Hadoop

Hadoop 是一套儲存並處理 petabytes 等級資訊的雲端運算技術

Hadoop

- http://hadoop.apache.org
- Hadoop是Apache Top Level開發專案
- 目前主要由Yahoo!資助、開發與運用
- Facebook、Last.fm、Joost are also powered by Hadoop
- 創始者是Doug Cutting，參考Google Filesystem
- 以Java開發，提供HDFS與MapReduce API。
- 2006年使用在Yahoo內部服務中
- 已佈署於上千個節點。
- 處理Petabyte等級資料量。

動機 / 條件

Data > 1 TB

交互運算於大量CPU

使用Map / Reduce 框架

High-level applications written in MapReduce

Programmers don't worry about socket(), etc.

特色

巨量

擁有儲存與處理大量資料的能力

經濟

可以用在由一般PC所架設的叢集環境內

效率

籍平行分散檔案的處理以得到快速的回應

可靠

當某節點發生錯誤，系統能即時自動的取得備份資料以及佈署運算資源

Hadoop於yahoo的運作資訊

年份	日期	節點數	耗時（小時）
2006	四月	188	47.9
2006	五月	500	42
2006	十一月	20	1.8
2006	十一月	100	3.3
2006	十一月	500	5.2
2006	十一月	900	7.8
2007	七月	20	1.2
2007	七月	100	1.3
2007	七月	500	2
2007	七月	900	2.5

Sort benchmark, every nodes with terabytes data.

誰在用Hadoop ？(1)

- **Facebook**
  - 處理 internal log and dimension data sources for reporting/analytics and machine learning.
- **IBM**
  - Blue Cloud Computing Clusters
- **Journey Dynamics**
  - 用 Hadoop MapReduce 分析 billions of lines of GPS data 並產生交通路線資訊.
- **Krugle**
  - 用 Hadoop and Nutch 建構 原始碼搜尋引擎

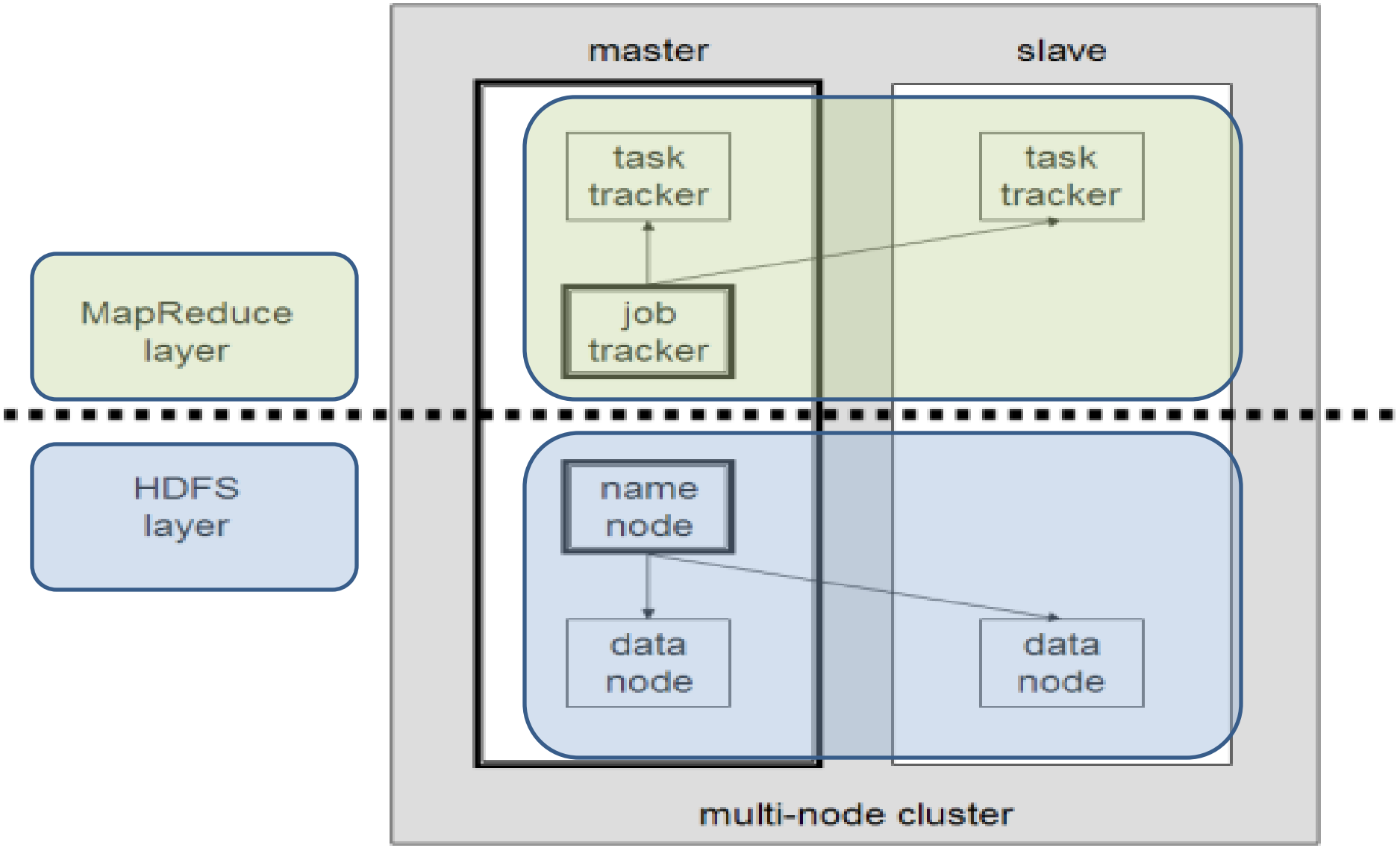
誰在用Hadoop ？(2)

- **SEDNS - Security Enhanced DNS Group**
  - 收集全世界的 DNS 以探索網路分散式內容.
- **Technical analysis and Stock Research**
  - 分析股票資訊
- **University of Nebraska Lincoln, Research Computing Facility**
  - 用Hadoop跑約200TB的CMS經驗分析
  - **緊湊渺子線圈**（**CMS**，**C**ompact **M**uon **S**olenoid）為瑞士歐洲核子研究組織CERN的大型強子對撞器計劃的兩大通用型粒子偵測器中的一個。

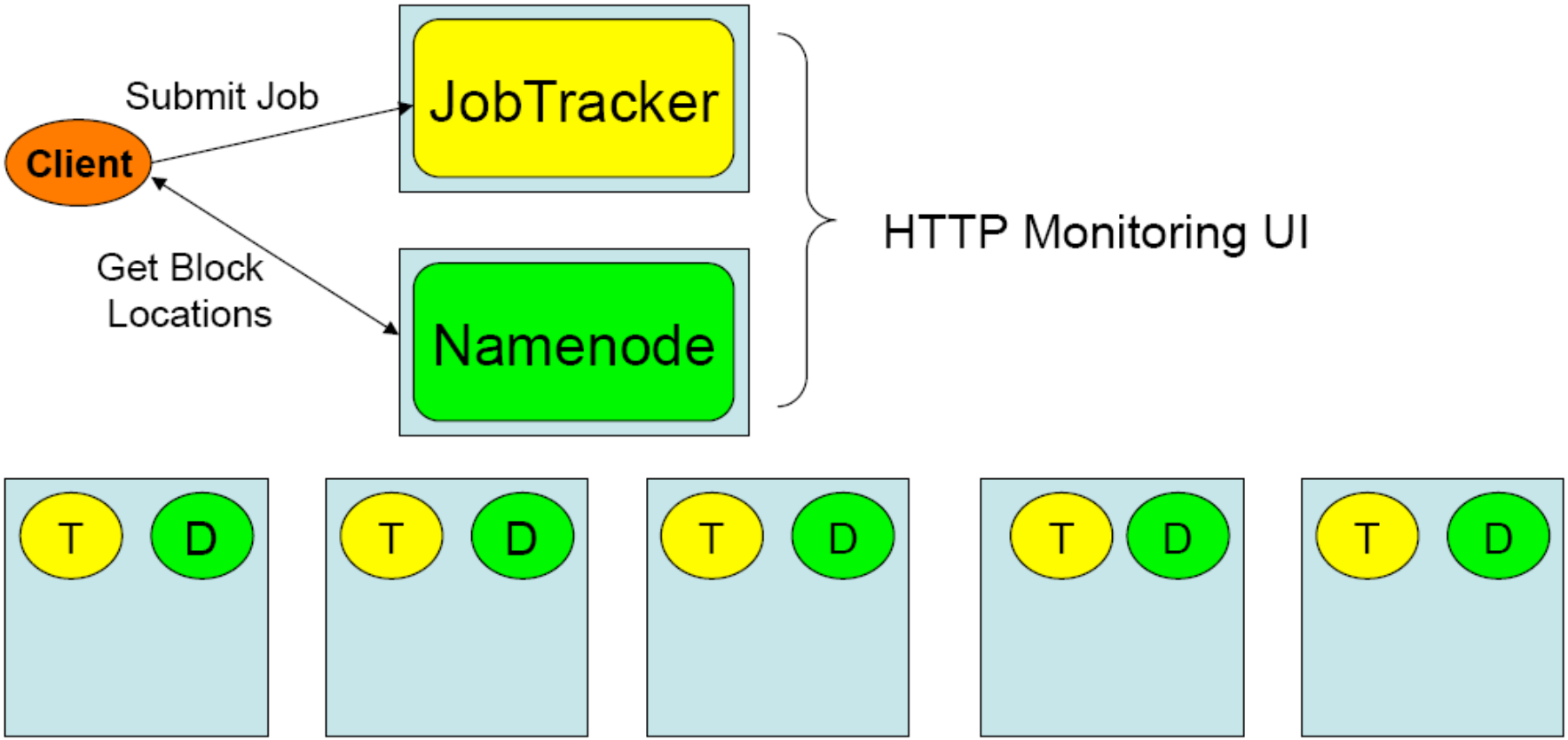
誰在用Hadoop？(3)

- **Yahoo!**
  - Used to support research for Ad Systems and Web Search
  - 使用Hadoop平台來發現發送垃圾郵件的殭屍網絡
- **趨勢科技**
  - 過濾像是釣魚網站或惡意連結的網頁內容

Hadoop 的主要架構



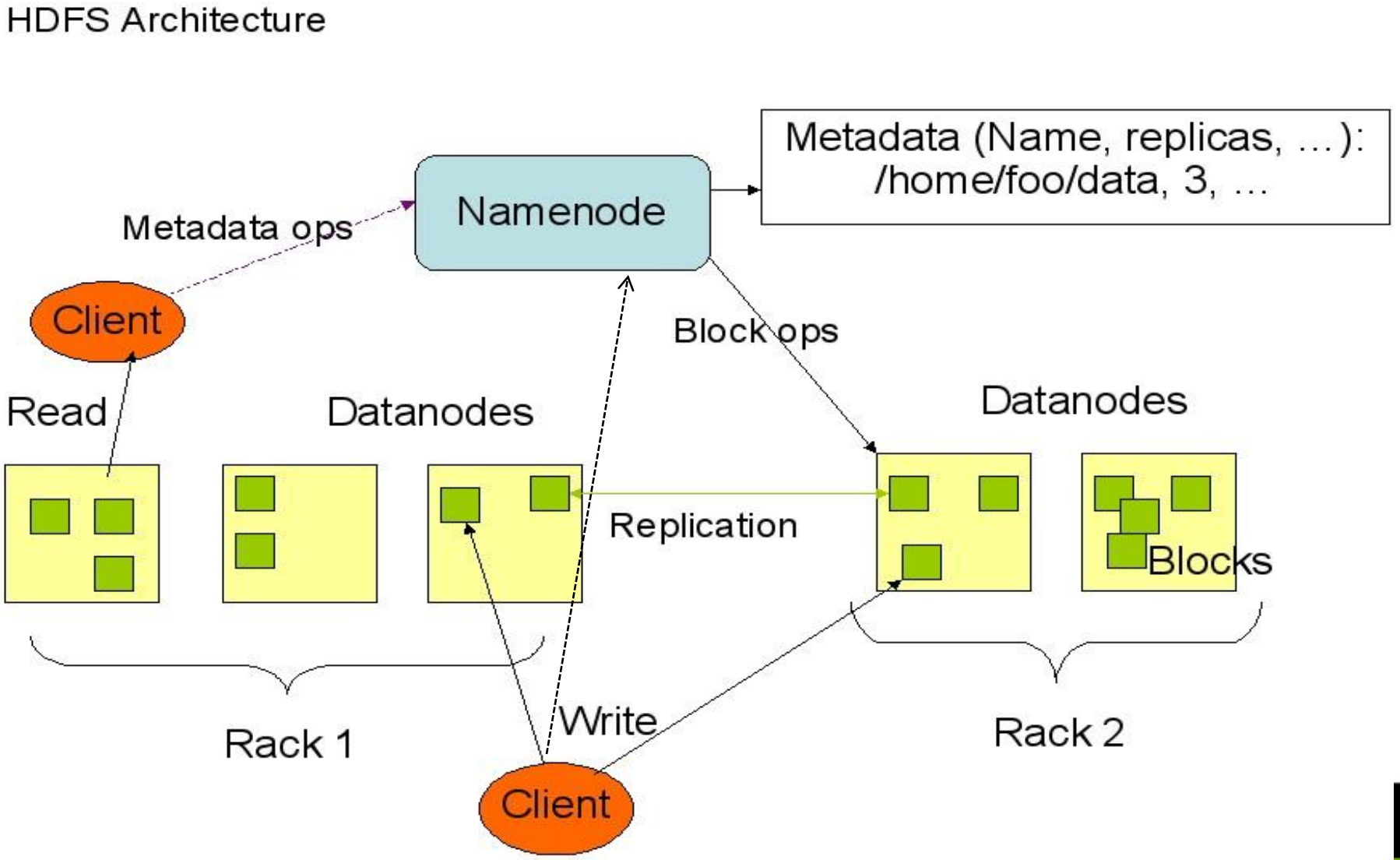
雲的Client 端


Submit Job
Client
JobTracker
Get Block
Locations
Namenode
HTTP Monitoring UI
T
D
T
D
T
D
T
D
T
D

HDFS 設計準則

- 檔案以區塊(block)方式儲存
  - 每個區塊大小遠比多數檔案系統都來得大(預設值為64MB)
- 透過複本機制來提高可靠度
  - 每個區塊至少備分到三台以上的DataNode
- 單一master (NameNode) 來協調存取及屬性資料(metadata)
  - 簡易的集中控管機制
- 沒有資料快取機制(No data caching)
  - 快取對於大資料集與串流讀取沒太大幫助
- 熟悉的介面，但客制化的API
  - 簡化問題；專注於分散式應用

Namenode管理HDFS資料

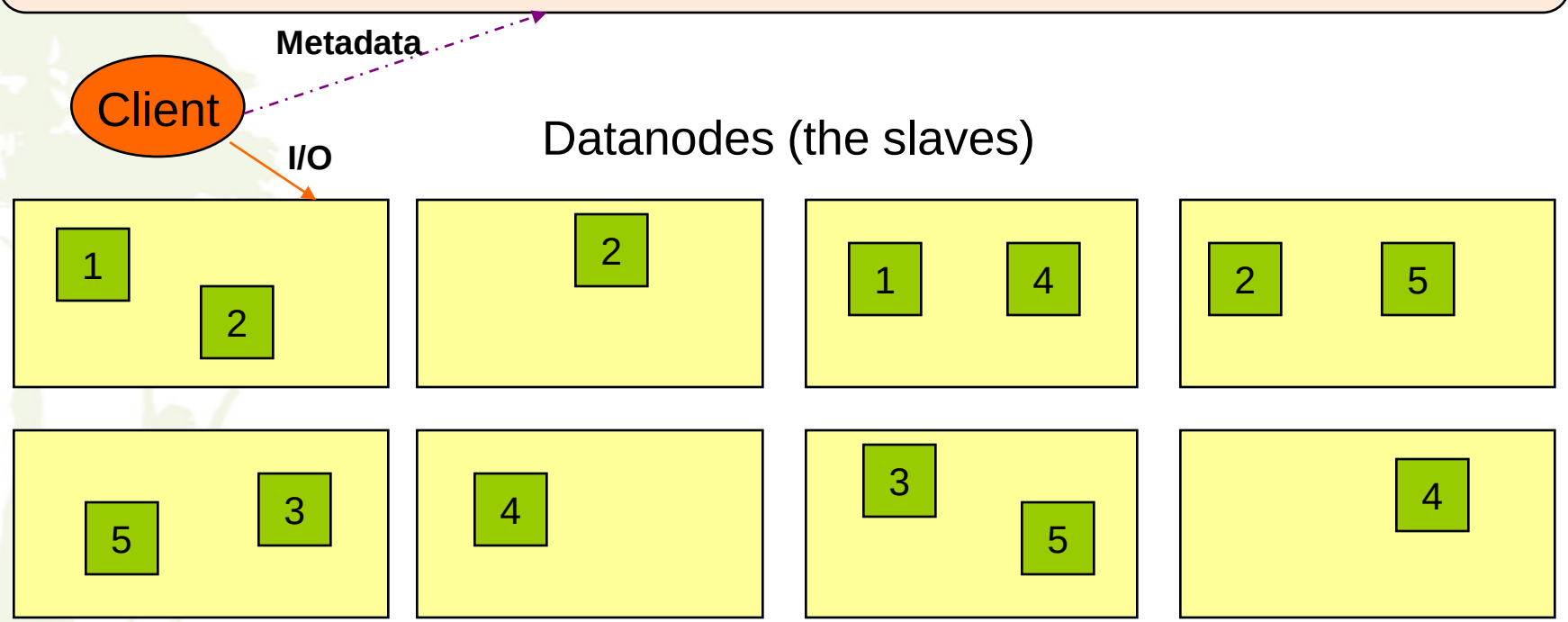


HDFS 的檔案讀寫

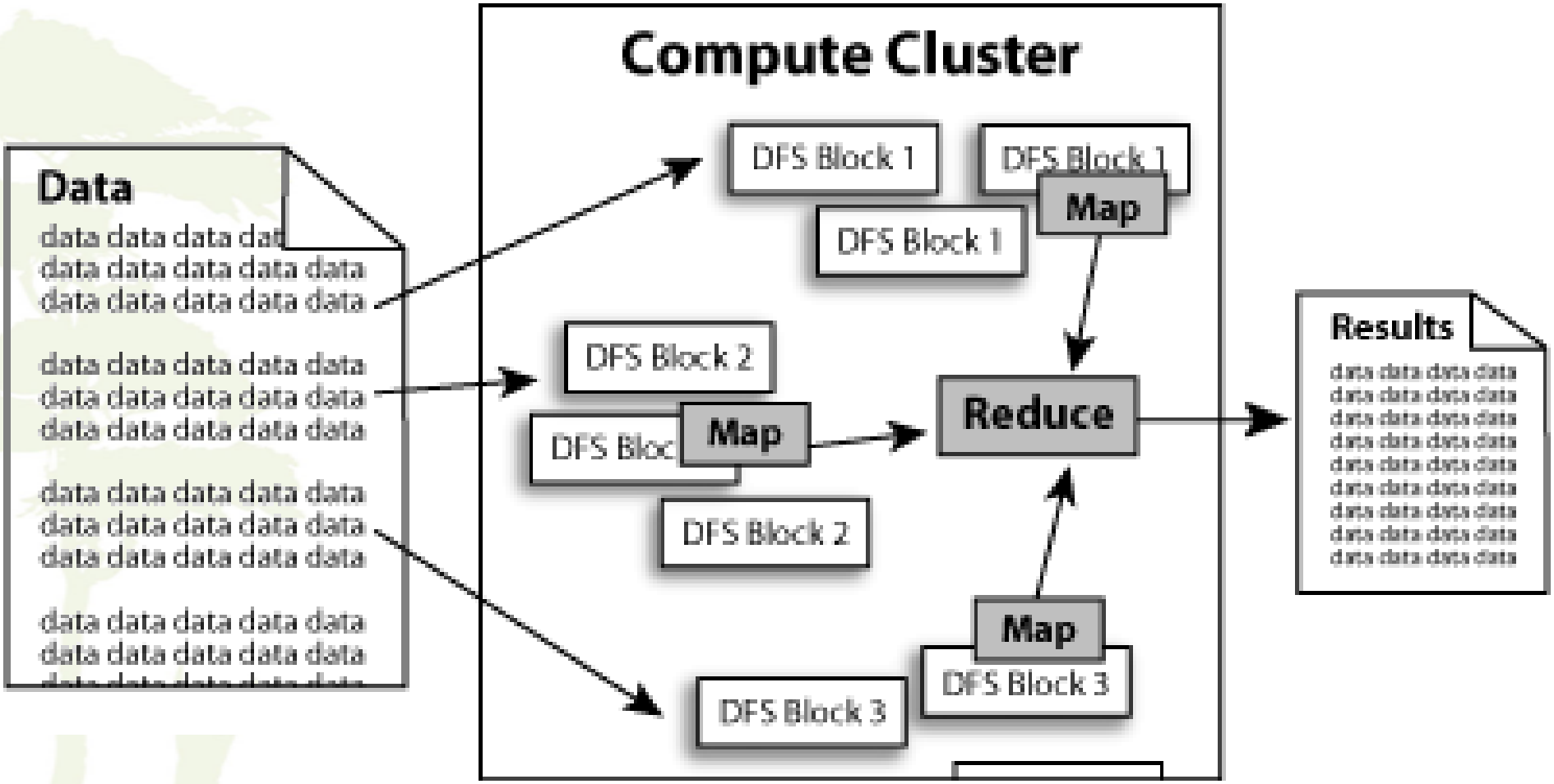
Namenode (the master)

檔案路徑 – 副本數 , 由哪幾個block組成

name:/users/joeYahoo/myFile - copies:2, blocks:{1,3}
name:/users/bobYahoo/someData.gzip, copies:3, blocks:{2,4,5}



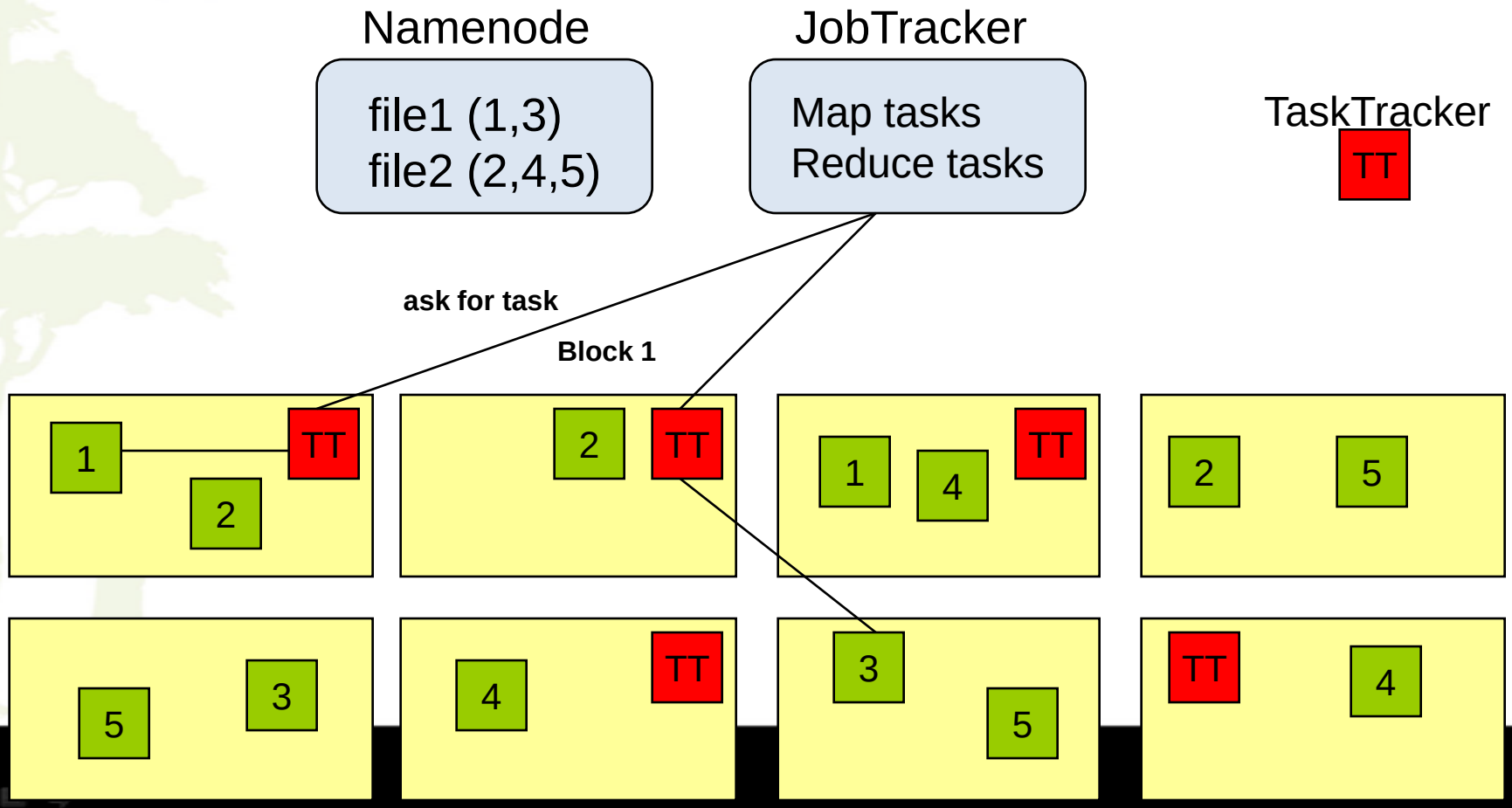
Map / Reduce Framework



MapReduce is a software framework to support distributed computing on large data sets on clusters of computers.

Jobs 搭配 HDFS 運算

- 可靠性：節點失效時讀取副本已維持正常運作
- 讀取效率：分散讀取流量（但增加寫入時效能瓶頸）



Map

- **One-to-one Mapper**

```
let map(k, v) =
Emit(k.toUpper(),
v.toUpper())
```

(“Foo”, “other”) → (“FOO”, “OTHER”)
(“key2”, “data”) → (“KEY2”, “DATA”)

- **Explode Mapper**

```
let map(k, v) =
foreach char c in v:
  emit(k, c)
```

(“A”, “cats”) → (“A”, “c”), (“A”, “a”),
(“A”, “t”), (“A”, “s”)

- **Filter Mapper**

```
let map(k, v) =
if (isPrime(v)) then
emit(k, v)
```

(“foo”, 7) → (“foo”, 7)
(“test”, 10) → (nothing)

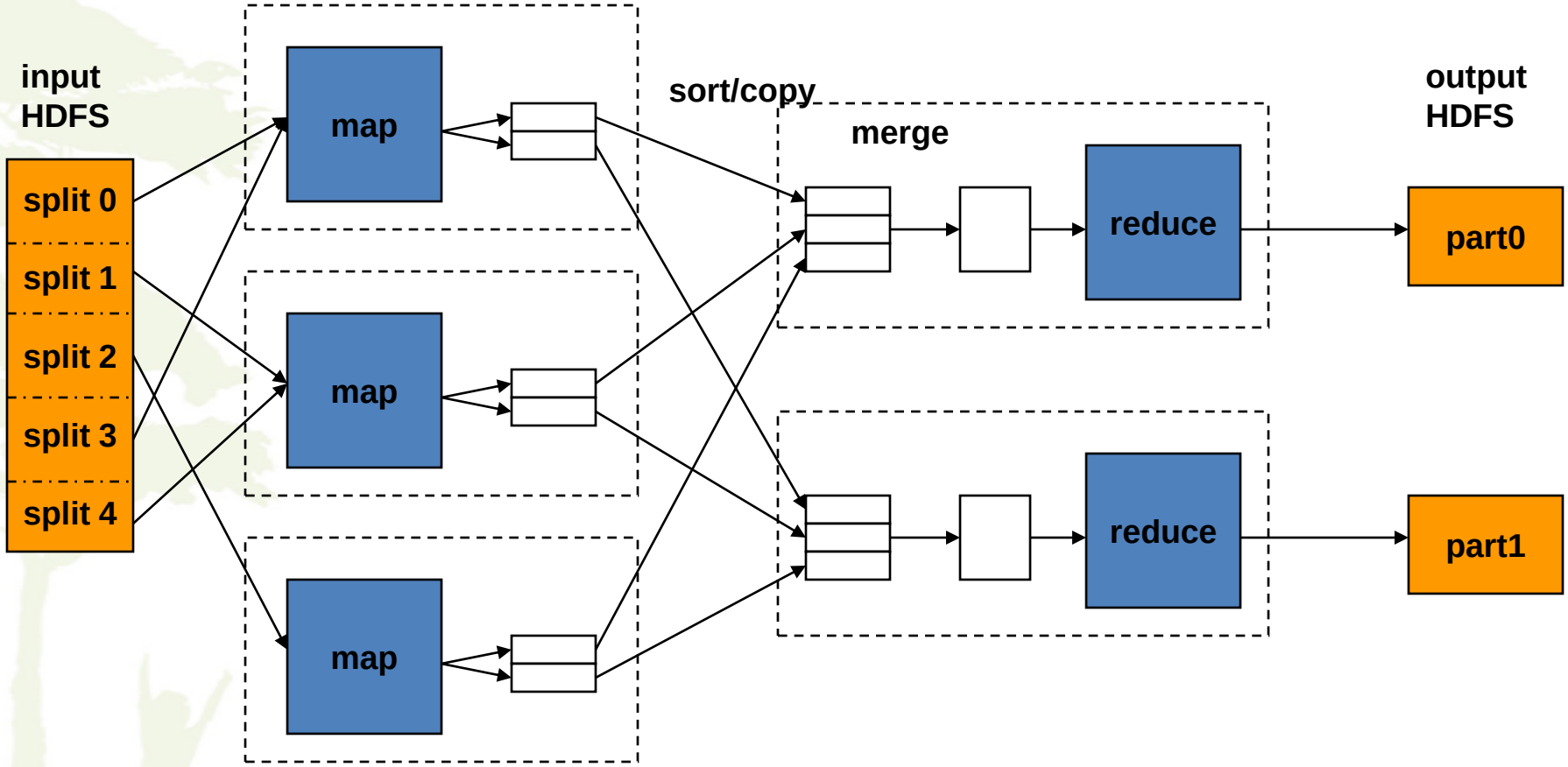
Reduce

Example: Sum Reducer

```
let reduce(k, vals) =
    sum = 0
    foreach int v in vals:
        sum += v
    emit(k, sum)
```

("A", [42, 100, 312]) → ("A", 454)
("B", [12, 6, -2]) → ("B", 16)

MapReduce 運作流程



JobTracker跟NameNode取得需要運算的blocks

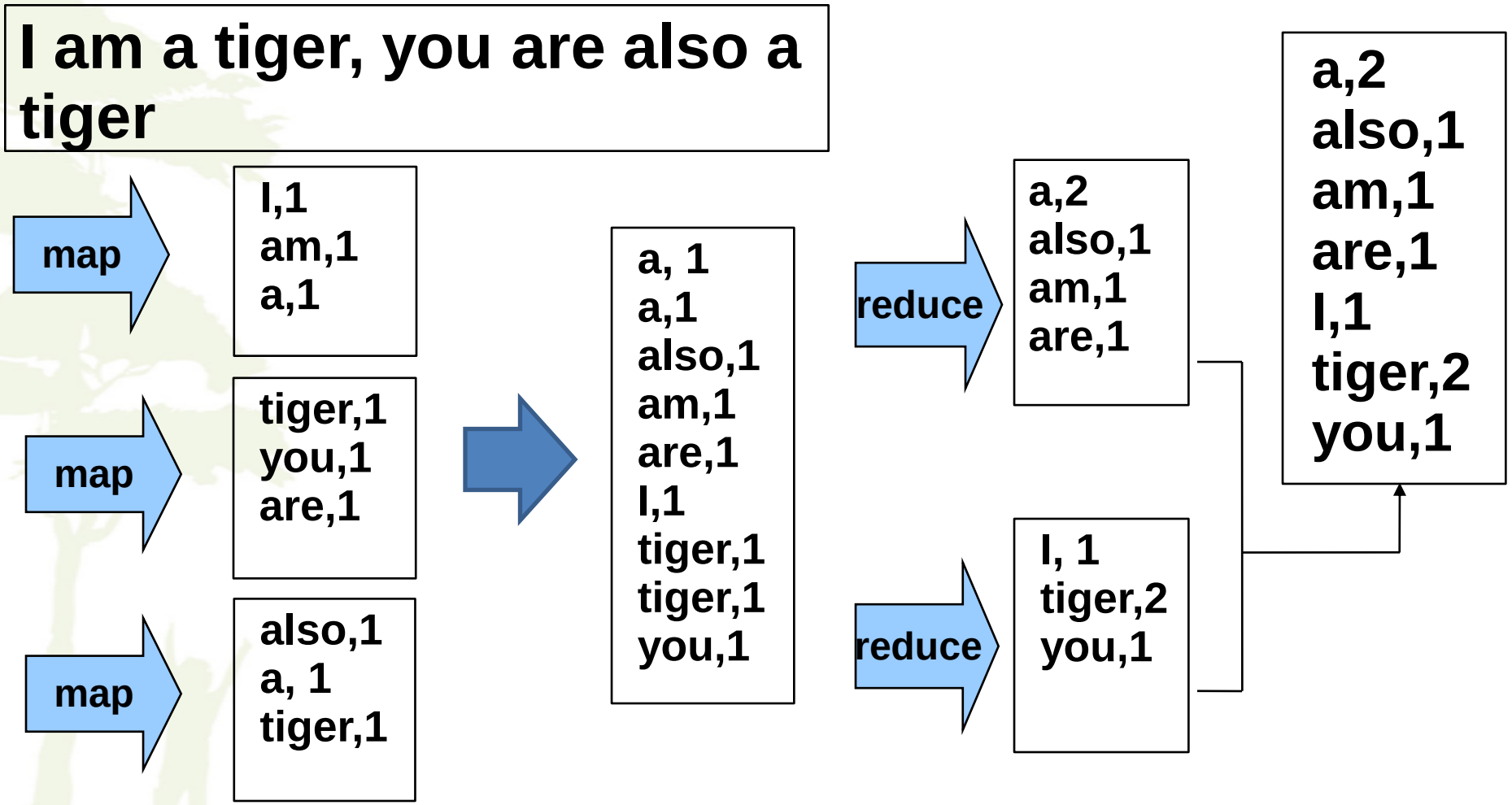
JobTracker選數個TaskTracker來作Map運算，產生些中間檔案

JobTracker將中間檔案整合排序後，複製到需要的TaskTracker去

JobTracker派遣TaskTracker作reduce

reduce完後通知JobTracker與Namenode以產生output

實例範例



JobTracker先選了三個Tracker做map

Map結束後，hadoop進行中間資料的整理與排序

JobTracker再選兩個TaskTracker作reduce

從 HADOOP 到搜尋引擎的應用

關鍵字 java 搜尋？

標題

用戶名

JAVA 技術手冊投影片 (PDF @ BDG @ HTTP) 1 2

搜索選項

SE (角色扮演)遊戲大合集遊戲支援java手機[MU、聯盤、無限@HTTP@22.7MB] 1 2

搜索方式

SE 268款游戲支援java手機[MU、mofile@HTTP@121MB] 1 2 3 4

全文搜索

JavaSc

主題 視

過濾出在資料庫內的所有資料中，有含此關鍵字內容的挑選方法

搜索時間

移除多

javaSC

Java深

JavaScript Cookbook(PDF @ MU @ HTTP)

每台電腦都需要的 Java Version 6 Update 17 最新版 1 2 3

> 學分兌換 - 福利社

搜索

搜尋？限制？

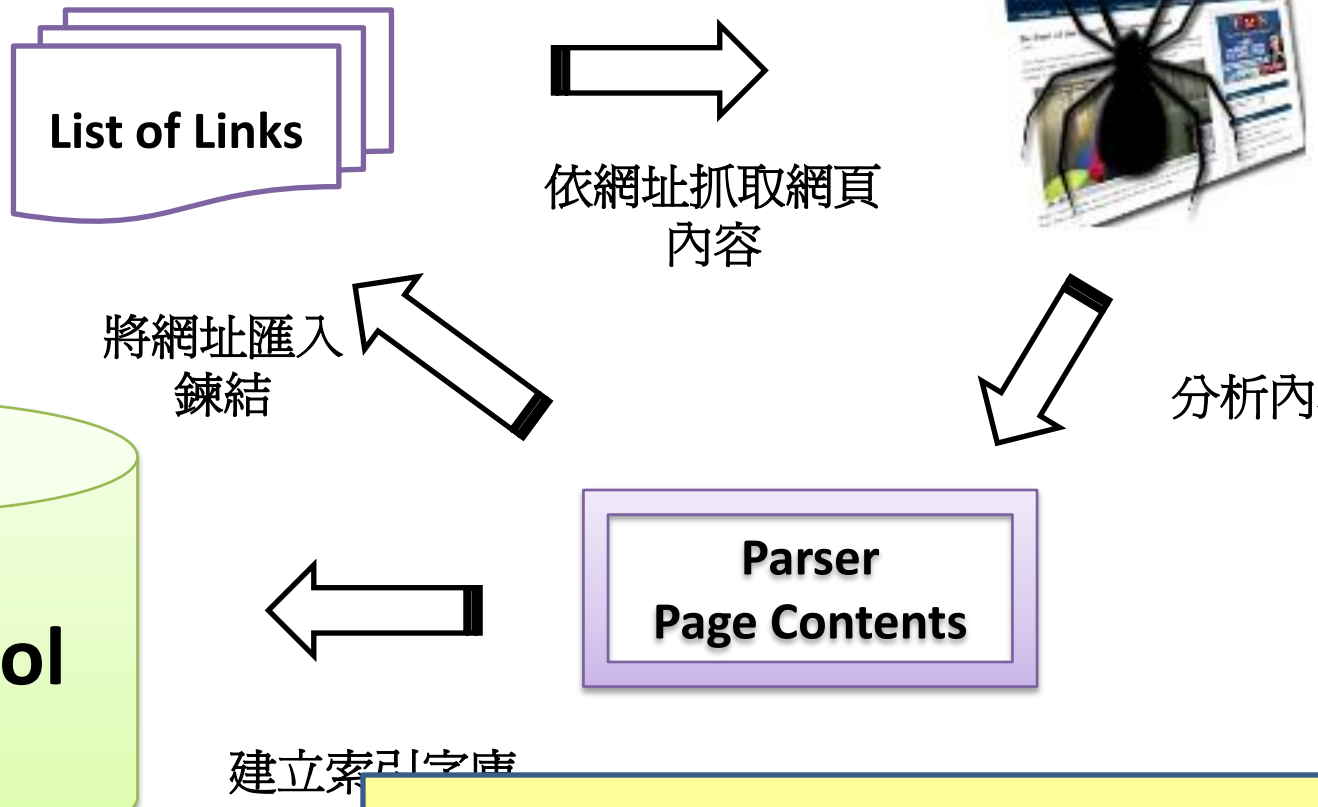
- 搜尋的範圍越大、資料越多，則會需要越多搜尋時間
- 僅限用資料庫架站
- 無法搜尋站外資料
- 無法搜尋附檔內容
- 搜尋 “Java 技術”，無法找到 “java 的技術手冊”

論壇的搜尋功能
＝
關連試資料庫的搜尋功能

全球網頁的搜尋引擎

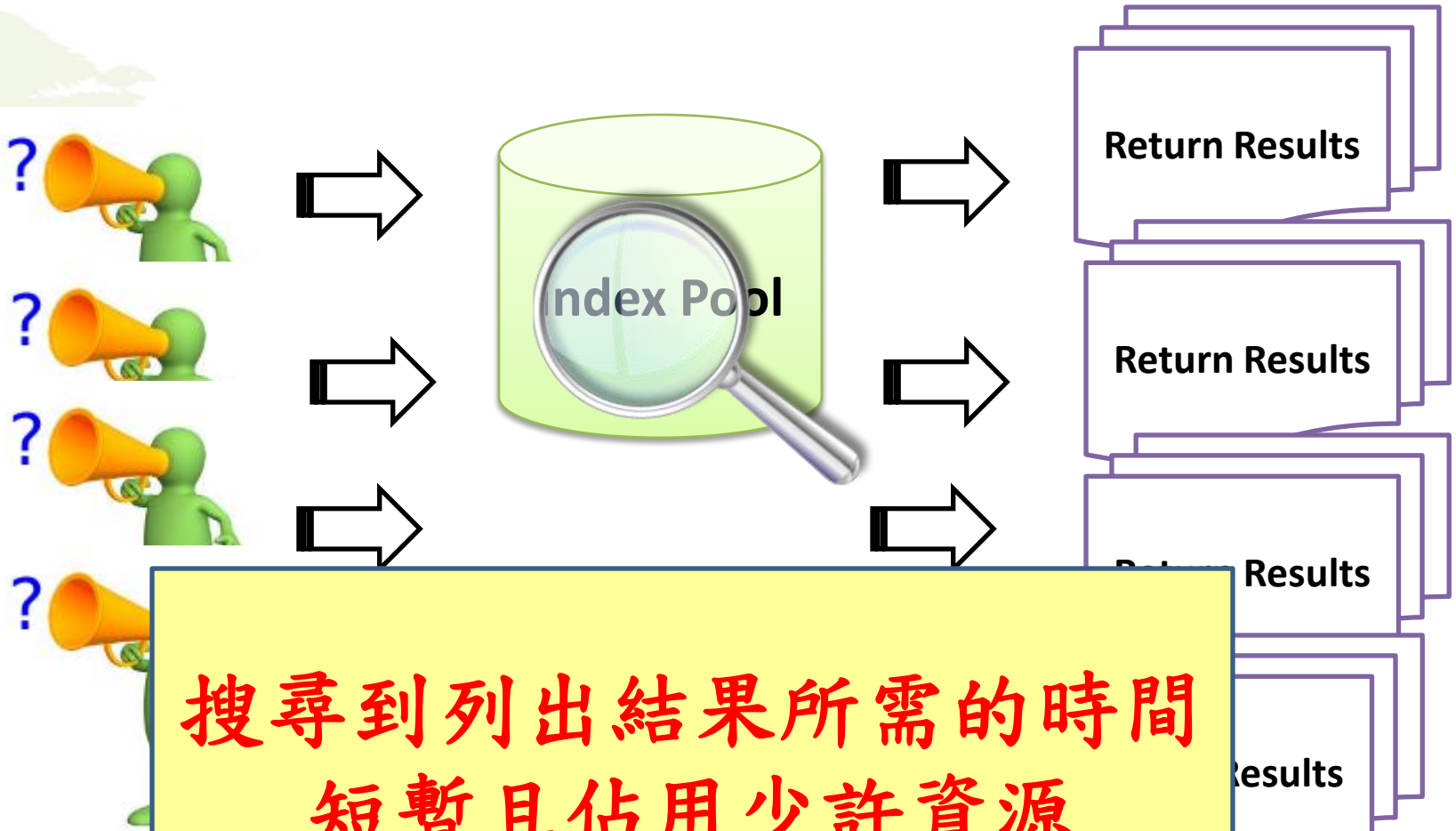
- 如 google, yahoo, bing , baidu 等，只要是開放的資料，有連結網址，就有機會被搜尋到。

搜尋引擎運作原理



建立索引字庫需要大量的運算資源與時間

搜尋引擎運作原理



搜尋到列出結果所需的時間
短暫且佔用少許資源

回到Hadoop

- **Doug Cutting**
- **Nutch**
  - 網路爬蟲 + 檔案格式分析+ 全文索引+ 分散式檔案系統 + mapreduce 運算架構
- **Nutch 0.8 版本之後**
  - 獨立出“分散式檔案系統 + mapreduce 運算架構”於 Hadoop 專案
- **Hadoop 0.20 版本之後**
  - 子項目：Core 、 Map / Reduce 、HDFS

Hadoop 一開始是開發來處理搜尋引擎工作的用途


http://localhost:50030/jobtracker.jsp
localhost Hadoop Map/Reduce ...
Nutch 與 Hadoop
job_201011251829_0041 NORMAL waue index-lucene thf4/indexes 100.00% 6 100.00% 1
job_201011251829_0042 NORMAL waue dedup 1: urls by time 100.00% 1 1 100.00% 1
job_201011251829_0043 NORMAL waue dedup 2: content by hash 100.00% 1 1 100.00% 1
job_201011251829_0044 NORMAL waue dedup 3: delete from index(es) 100.00% 1 1 100.00% 1
job_201011251829_0045 NORMAL waue inject url 100.00% 2 2 100.00% 1
job_201011251829_0046 NORMAL waue crawldb thf5/crawldb 100.00% 1 1 100.00% 1
job_201011251829_0047 NORMAL waue generate: select from thf5/crawldb 100.00% 1 1 100.00% 1
job_201011251829_0048 NORMAL waue generate: partition thf5/segments /20101126161446 100.00% 1 1 100.00% 2
job_201011251829_0049 NORMAL waue fetch thf5/segments /20101126161446 100.00% 2 2 100.00% 1
job_201011251829_0050 NORMAL waue crawldb thf5/crawldb 100.00% 4 4 100.00% 1
job_201011251829_0051 NORMAL waue generate: select from thf5/crawldb 100.00% 2 2 100.00% 1
job_201011251829_0052 NORMAL waue generate: partition thf5/segments /20101126161721 100.00% 2 2 100.00% 2
job_201011251829_0053 NORMAL waue fetch thf5/segments /20101126161721 100.00% 2 2 100.00% 1
job_201011251829_0054 NORMAL waue crawldb thf5/crawldb 100.00% 4 4 100.00% 1
job_201011251829_0055 NORMAL waue linkdb thf5/linkdb 100.00% 3 3 100.00% 1
job_201011251829_0056 NORMAL waue index-lucene thf5/indexes 100.00% 10 10 100.00% 1
job_201011251829_0057 NORMAL waue dedup 1: urls by time 100.00% 1 1 100.00% 1
job_201011251829_0058 NORM
job_201011251829_0059 NORM
建立索引字庫

Nutch 與 Hadoop

- inject url
- crawldb JOBNAME /crawldb
- generate: select from JOBNAME /crawldb
- generate: partition JOBNAME /segments/20101126161446
- fetch JOBNAME /segments/20101126161446
- crawldb JOBNAME /crawldb
- linkdb JOBNAME /linkdb
- index-lucene JOBNAME /indexes
- dedup 1: urls by time
- dedup 2: content by hash
- dedup 3: delete from index(es)

Nutch 與 Tomcat

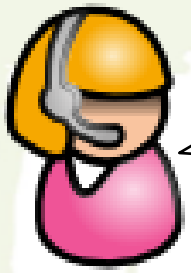
與Hadoop無直接相關

什麼是雲端運算啊？可以個簡單的定義嗎？

What is Cloud Computing ?

雲端運算怎麼聽起來要買一些新硬體、新軟體啊？

Is it about buying NEW Hardware and Software?



雲端運算可能只是拿來振興經濟的幌子吧？

Is it a trap to another bubble economy ?

我聽你們在那裡講五四三.....

Cloud Computing is as simple as 5..4..3..2..1...

原來搜尋引擎大致的原理長這樣！
不過我覺得現有的搜尋引擎已經很好用了

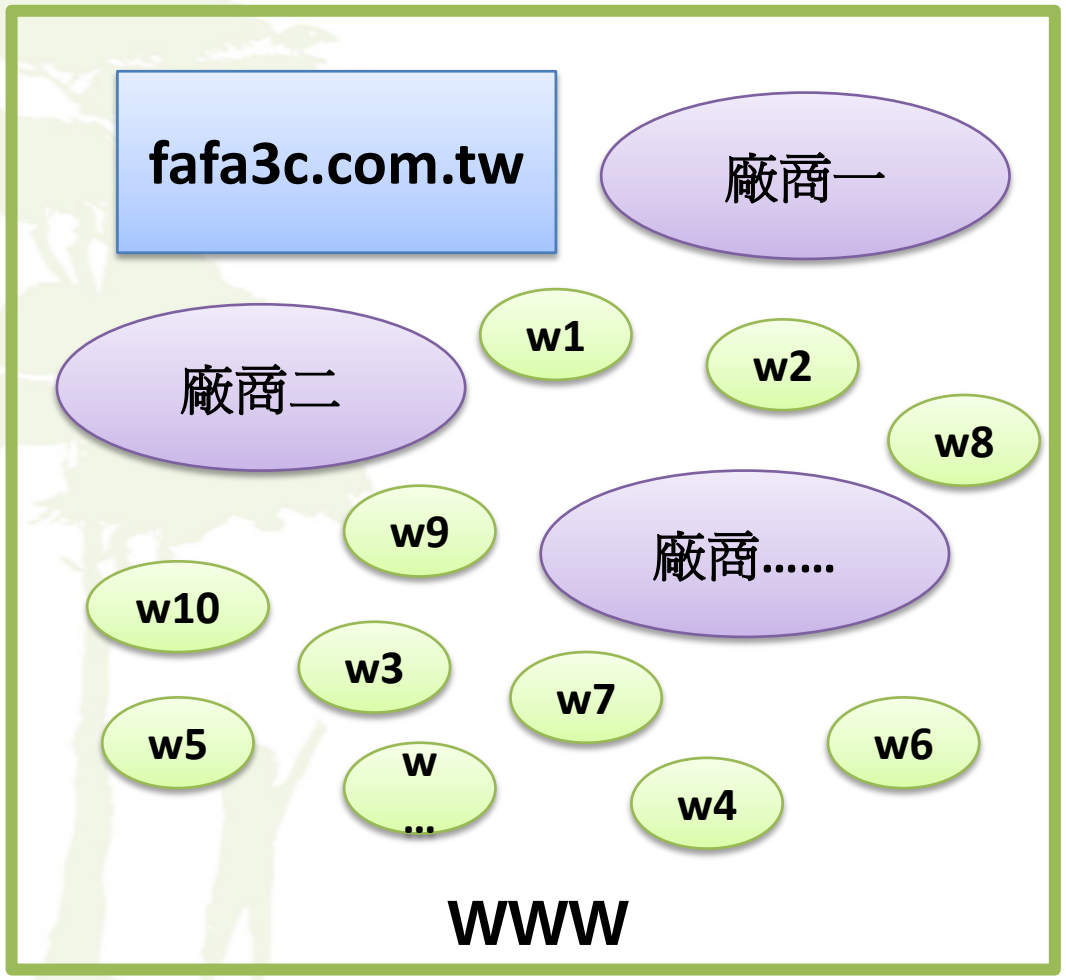
對阿，
為何要自己建搜尋引擎？

- 原因1: 建立防火牆內的搜尋
- 原因2: 減少廣告、不對的內容、啦里拉雜的資訊

實用案例

- 發發擁有一家專賣3C用品的貿易公司，也自己建立了一個貨品非常完整的網站，但是卻沒有自己網站的搜尋引擎
- 客戶或發發每次想找自己網站上面是否有該商品或此商品的完整型錄，都用一般的搜尋引擎來找，卻發現：需要的網頁通常埋沒在太多阿里不達的資訊海裡
- 於是發發苦思如何讓自己要的資訊 在搜尋引擎的結果排列在最前面

實用案例



Google 台灣

YAHOO! 奇摩

Result1 ：順x3C

Result2：就感心專賣

Result3：燦X3C

Result4：產品資訊

Result5：山寨大王電子專賣

.

.

.

.

.

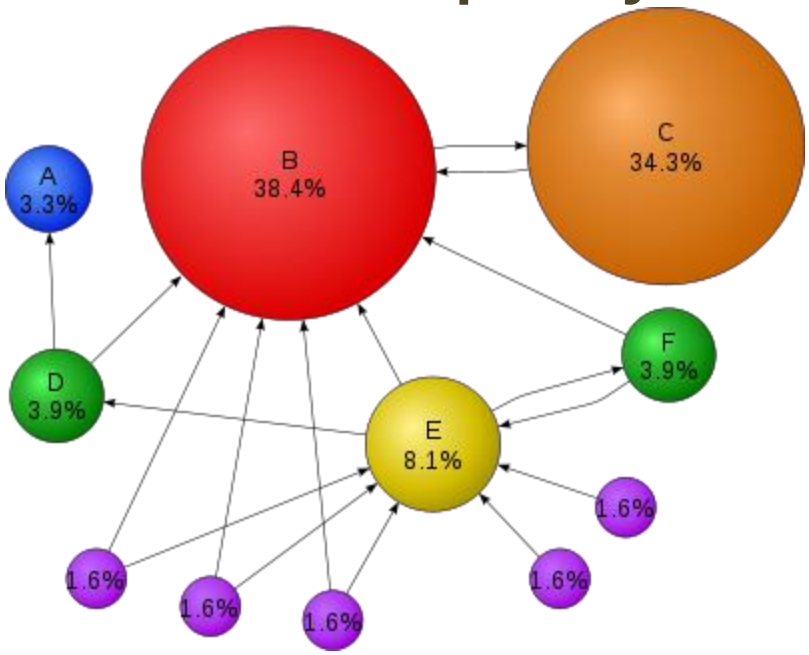
.

Search Results

實用案例

- 解決方法一：提昇自己網站的能見度
  - Search Engine Optimize (SEO) Tool
  - How Search Engines Rank Web Pages
    - Location, Location, Location...and Frequency

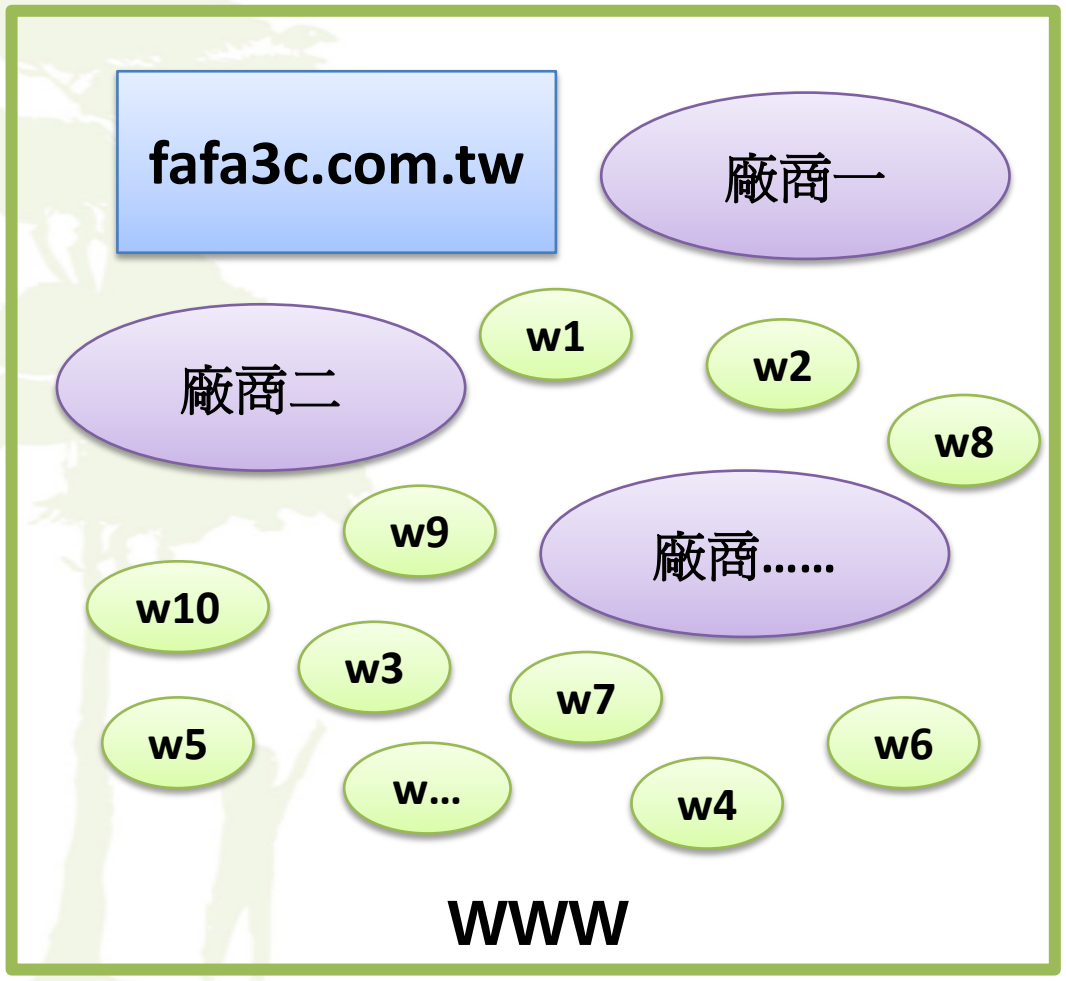
Mathematical **PageRanks** (out of 100) for a simple network (PageRanks reported by Google are rescaled logarithmically). Page C has a higher PageRank than Page E, even though it has fewer links to it; *the link* it has is of a much higher value. A web surfer who chooses a random link on every page (but with 15% likelihood jumps to a random page on the whole web) is going to be on Page E for 8.1% of the time. (The 15% likelihood of jumping to an arbitrary page corresponds to a damping factor of 85%.) Without damping, all web surfers would eventually end up on Pages A, B, or C, and all other pages would have PageRank zero. Page A is assumed to link to all pages in the web, because it has no outgoing links.



實用案例

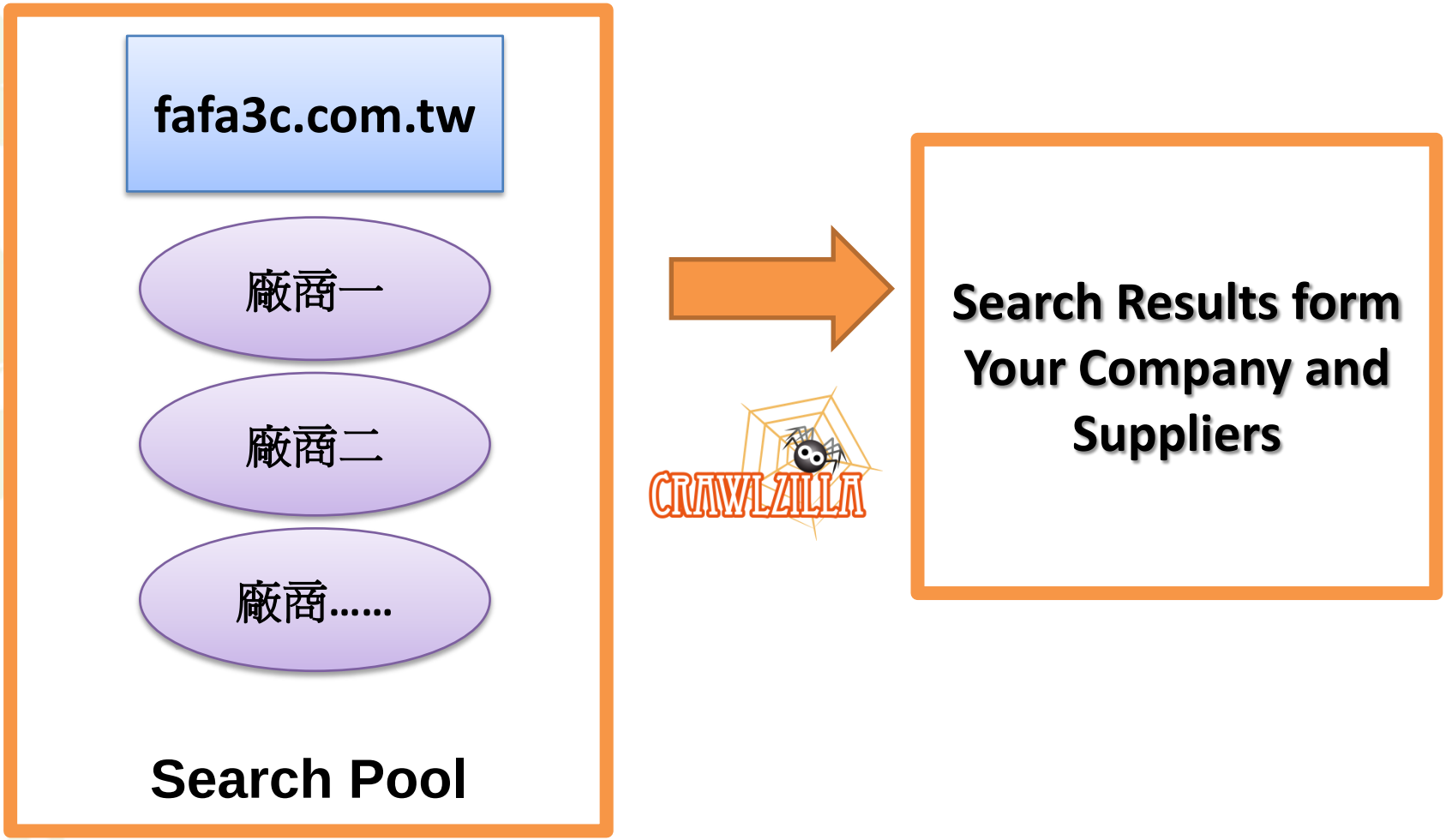
- 解決方法二：買關鍵字廣告
  - 每年都要花錢並且被各搜尋引擎業者制約
- 解決方法三： 用 關鍵字+ site: 發發.3c.com.tw 限制搜尋範圍
  - 但某些產品資訊是在製造商的網站上
- 如何解決?

透過Crawlzilla 提升搜尋品質!?
(以3C網站為例)



Search Pool

透過Crawlzilla 提升搜尋品質!?...(續)



用Nutch 建立自己的搜尋引擎

- 自由軟體
- 高效率
- 支援多種檔案格式
- 持續維護

- 安裝設定不易
- 指令操作
- 無索引庫資訊
- 中文不友善
- 網頁伺服器額外設定

- **Crawlzilla**：提供簡單安裝及操作管理介面，輕鬆建立搜尋引擎的自由軟體專案

你不需要懂Nutch，
你只需要Crawlzilla 懂你

- 專案網站
  - 中文：http://crawlzilla.info/
  - 英文：http://sf.net/p/crawlzilla

中文專案首頁

http://crawlzilla.info/

crawlzilla

國家高速網路與計算中心-自由軟體實驗室

Search projects

Project Home | Wiki | Issues | Source | Administer

Summary | Updates | People

Tip: Project owners, see our Getting Started guide for steps to configure your project. hide

Crawlzilla 輕鬆建立搜尋引擎的自由軟體

Visit The English Version

【Crawlzilla 簡介】

Crawlzilla 是一個 **「開源碼的叢集式搜尋引擎建制和管理工具」**，它支援了多種檔案格式(html、pdf、word...)的搜尋，並提供搜尋引擎的管理(爬取設定管理、叢集節點管理、索引庫管理)。主要目的是讓使用者能更輕鬆地建置自己專屬的搜尋引擎。

若您想對 Crawlzilla 有更親密一點的接觸，歡迎用您的滑鼠大力點選以下連結(大哉問、特色和操作畫面)。

Star this project

Activity: High

Code license:
Apache License 2.0

Labels:
Nutch, SearchEngine, Hadoop, CloudComputing, crawlzilla

External links:
下載Crawlzilla專案

Feeds:
Project feeds

Owners:
waue0920, goldjay1231, shunfa

Committers:
rider.tu, jazzwang.tw

People details »

英文專案首頁

http://sf.net/p/crawlzilla

sourceforge

crawlzilla

HOME | WIKI | DOWNLOADS | DISCUSSION | SVN | LINK

crawlzilla

Crawlzilla is a cluster-based search engine deployment tools. It helps user to build search engine in your cluster, and offers management mechanism (such as: cluster management, crawl management, index pool management...).

Documents:

- What is Crawzilla
- Installation
- Usage
- Disscusion
- Bug report

Download:

“搜尋”超越“搜尋”！

- 簡化安裝、方便操作、效能UP
  - 目前開源碼的搜尋引擎專案，(1)安裝過於繁瑣、(2)操作不夠友善、(3)單機版
- 中文搜尋的支援
- 圖形介面管理與操控
- 同時並存多個搜尋引擎
- 開發更專注

Crawlzilla 的開發項目

一、多種專案的架構整合

二、單機/叢集 並用的安裝流程

三、管理者管理介面

四、使用者介面的MVC 架構

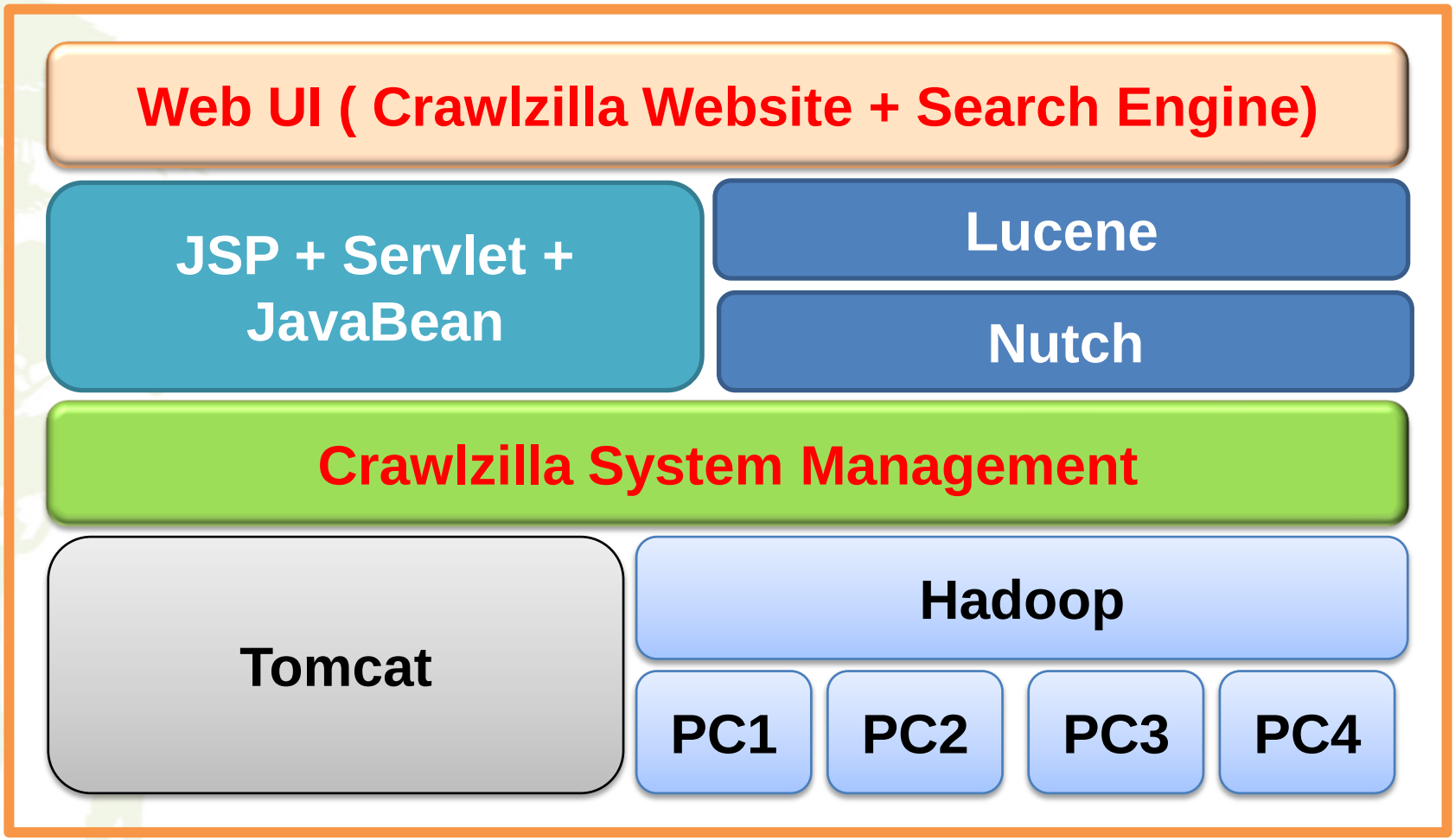
五、核心程式改良

六、協同開發與自動化生成安裝檔

七、更多功能

八、Feature Works

Crawlzilla 架構



	平行運算編程工具和分散式檔案系統，以Java實做Google大規模資料的開源碼專案
MapReduce	用於大規模資料平行運算的軟體架構
HDFS	具有高擴充、高容錯、低成本的大規模資料檔案系統
	基於Lucene Java 的開源碼搜尋引擎
	用於全文檢索和搜尋的開放源碼程式庫
	支援Java的輕量級網頁伺服器

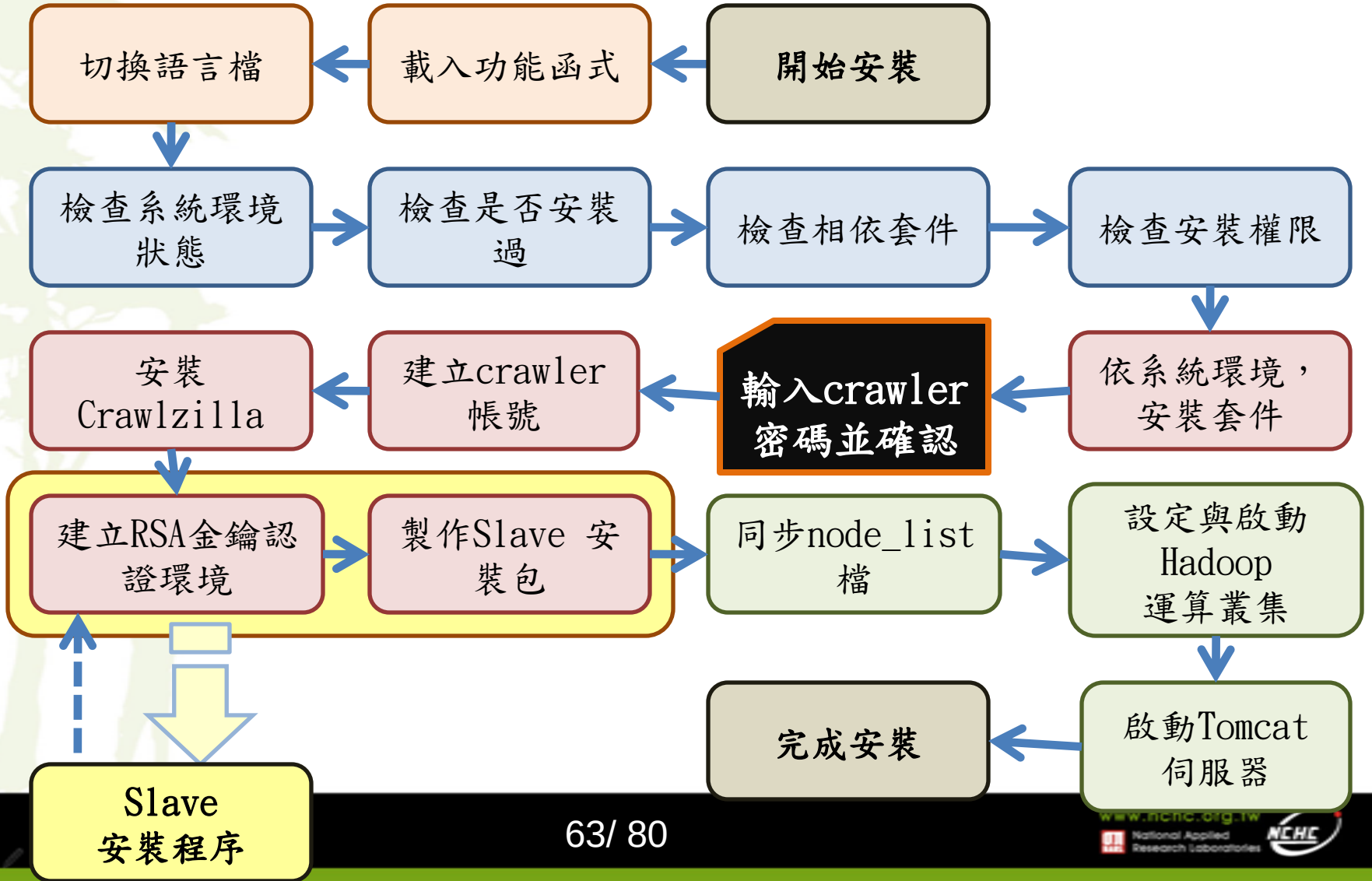
自行開發的介面

	JSP	Shell
功能	使用者UI	管理者UI
安全設定	網頁密語搭配 Session 設定	crawler密碼搭配 rsa 公私鑰密碼系統
雲端概念	任何有瀏覽器的裝置皆可操作	有ssh -client的終端機介面
軟體架構	MVC	模組化設計
多國語系	i18n	語言參數設定檔
	依系統環境自動挑選預設語言	

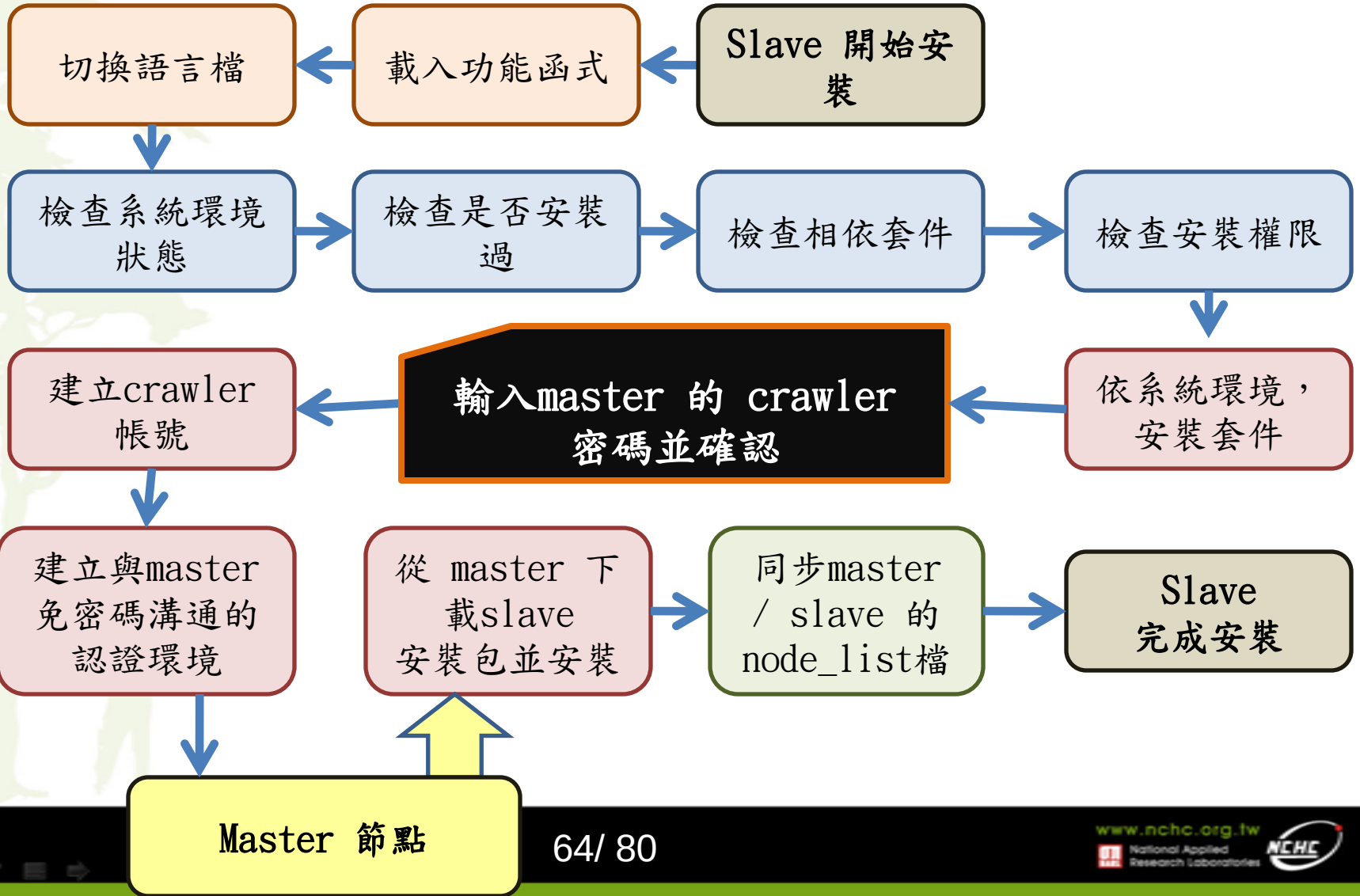
二、安裝步驟

- 使用者：
  - 下載→解壓縮→執行 ./install 指令→設定密碼→確認→完成
- 系統：
  - 自動完成系統判斷、套件相依、認證金鑰、套件安裝、環境設定、叢集設定等步驟

Master 安裝流程

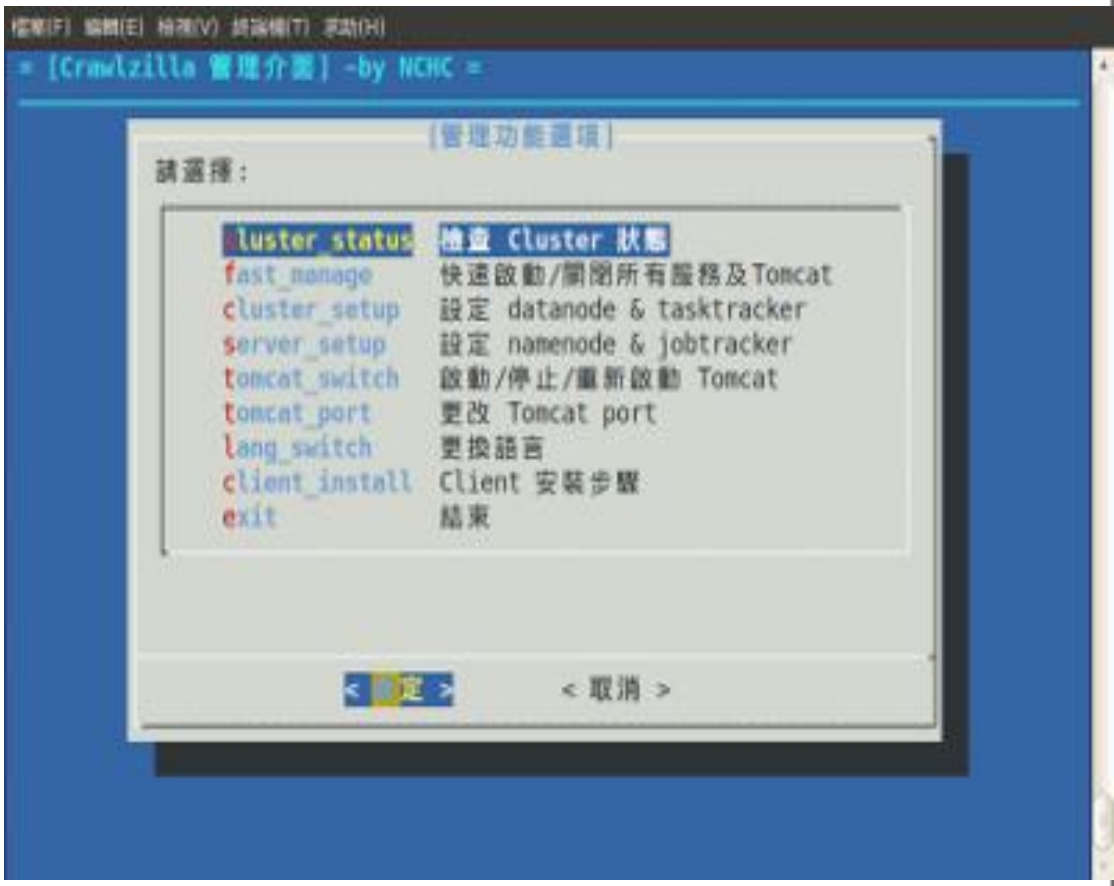

開始安裝
載入功能函式
切換語言檔
檢查系統環境狀態
檢查是否安裝過
檢查相依套件
檢查安裝權限
依系統環境，安裝套件
輸入crawler密碼並確認
建立crawler帳號
安裝Crawlzilla
建立RSA金鑰認證環境
製作Slave 安裝包
同步node_list檔
設定與啟動Hadoop運算叢集
啟動Tomcat伺服器
完成安裝
Slave安裝程序

Slave 安裝程序


Slave 開始安裝
載入功能函式
切換語言檔
檢查系統環境狀態
檢查是否安裝過
檢查相依套件
檢查安裝權限
依系統環境，安裝套件
輸入master 的 crawler 密碼並確認
建立crawler帳號
建立與master免密碼溝通的認證環境
從 master 下載slave 安裝包並安裝
同步master / slave 的 node_list檔
Slave 完成安裝
Master 節點

三、系統管理介面

- 檢查cluster 狀態
- 啟動、設定Tomcat伺服器
- 啟動Hadoop的運算、儲存節點
- 變換語言
- 提示安裝訊息



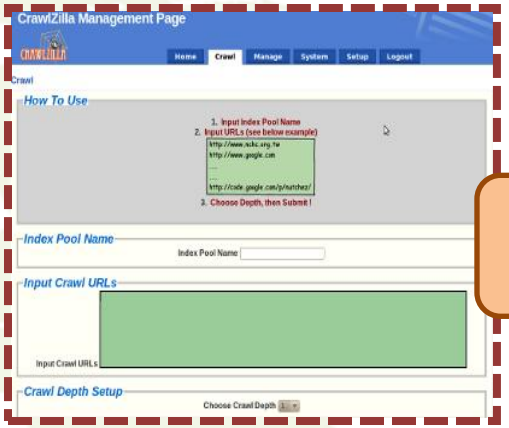
叢集間的溝通

- **Master Control ： Master可以動態地啟動 / 停止任一節點上的運算或儲存服務**
- **Node List ：一旦某一slave 被執行 install / remove ，node list都會被更新並且同步到每一台機器上**
- **Asymmetric-Key Encryption：利用RSA Private / Public Key ，使得節點之間的溝通安全無礙**

四、使用者介面的MVC 架構

Model 2

驅動與設定網頁爬取程序



Servlet

Control

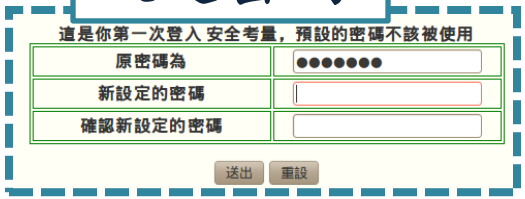
JavaBean

Model

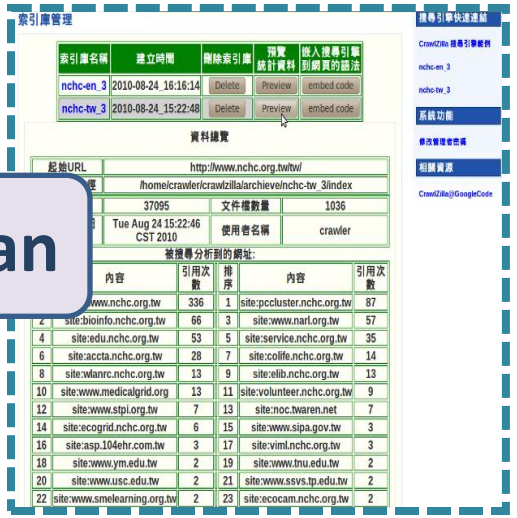
View

JSP

設定密碼

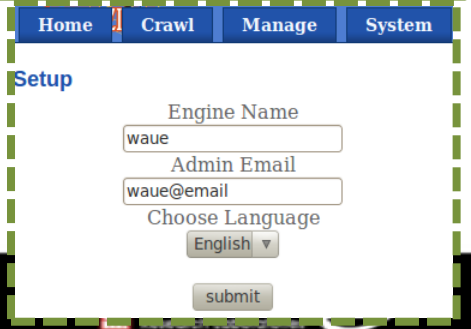


擷取 Lucene索引資料庫

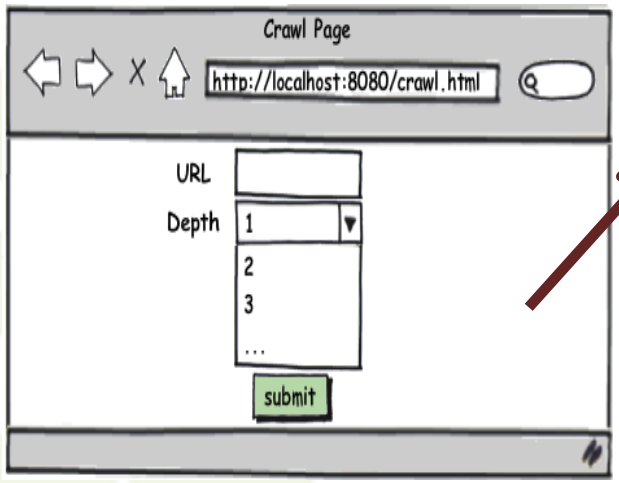


系統介面整合與頁面呈現

i18N 語言設定



網頁爬取程序



Servlet

Java Beans

JSP

Nutch

Lucene Index DB

索引庫名稱	建立時間	爬取深度	爬取時間	刪除索引庫	預覽統計資料
www2_iso_4	2010-11-29_11:16:38	4	4:4:4	Delete	Preview

資料總覽

起始URL			
本機索引路徑			
總共文字數	0	文件檔數量	
索引庫更新日期		使用者名稱	

索引庫更新日期	Tue Aug 24 15:22:46 CST 2010	使用者名稱	crawler

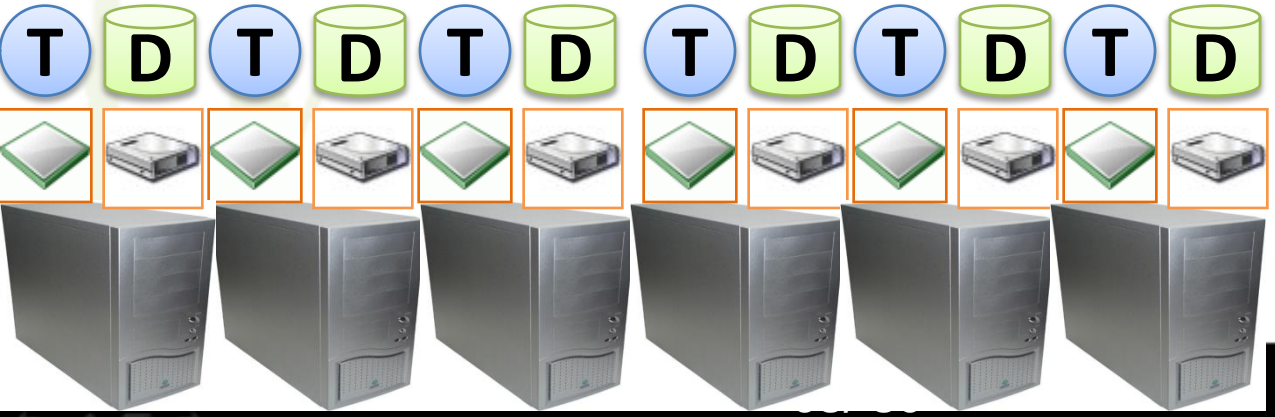
被搜尋分析到的網址:

排序	內容	引用次數	排序	內容	引用次數
0	site:www.nchc.org.tw	336	1	site:pccluster.nchc.org.tw	87
2	site:bioinfo.nchc.org.tw	66	3	site:www.narl.org.tw	57
4	site:edu.nchc.org.tw	53	5	site:service.nchc.org.tw	35
6	site:accta.nchc.org.tw	28	7	site:colife.nchc.org.tw	14
8	site:wlanrc.nchc.org.tw	13	9	site:elib.nchc.org.tw	13
10	site:www.medicalgrid.org	13	11	site:volunteer.nchc.org.tw	9
12	site:www.stpi.org.tw	7	13	site:noc.twaren.net	7
14	site:ecogrid.nchc.org.tw	6	15	site:www.sipa.gov.tw	3
16	site:asp.104ehr.com.tw	3	17	site:viml.nchc.org.tw	3
18	site:www.ym.edu.tw	2	19	site:www.tnu.edu.tw	2
20	site:www.usc.edu.tw	2	21	site:www.ssvs.tp.edu.tw	2
22	site:www.smelearning.org.tw	2	23	site:ecocam.nchc.org.tw	2

Hadoop

JobTracker

NameNode



T TaskTracker: 工作執行者

D DataNode: 資料儲存節點

中文分詞-解說

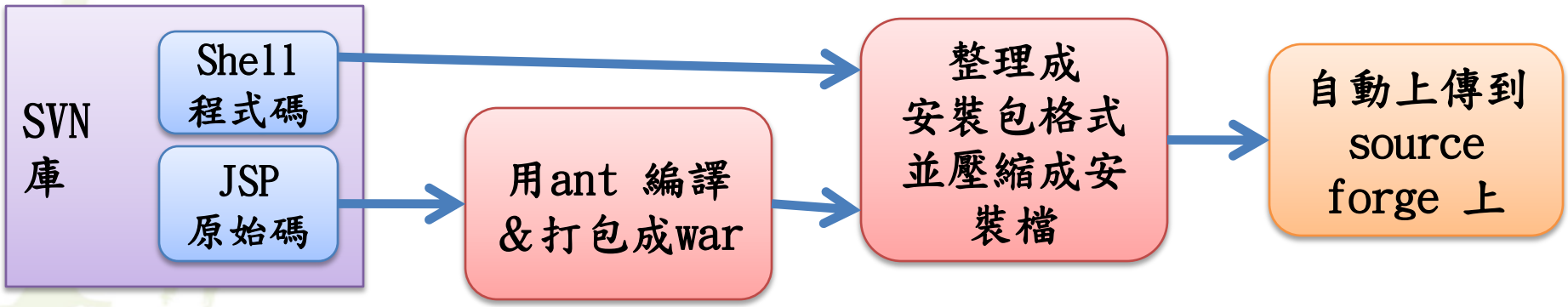
- 原本的分析引擎 Nutch 並無中文分詞支援
- “今天天氣真好”，若搜尋"氣真“
  - 沒有中文分詞的結果為：1筆 => “今”,“天”,“氣”,“真”,“好”
  - 有中文分詞：0筆 (因為詞庫為今、今天、天、天天、天氣、真、真好、好)
- 將Crawlzilla 的 Nutch 加入中文分詞 功能

中文分詞- 方法

- 將IK-Analyzer包入Crawlzilla 內
  - IKAnalyzer ＝ java、輕量、開源的中文分詞工具包
  - 具有60萬字/秒的高速處理能力
  - 英文字母（IP地址、Email、URL）、數位（日期，常用中文數量詞，羅馬數字，科學計數法）
  - 正體簡體中文詞彙（姓名、地名處理）等分詞處理
- 整合Lucene 與 IK-Analyzer
- 改寫搜尋引擎核心Nutch的分析程式
- 重新編譯出新的Nutch語意分析核心

六、自動化生成安裝檔

- 原始程式碼：位於 協同開發的svn 庫中
  - 兩種程式語言，不同開發工具，因此svn 上有 shell/ 與 jsp/ 資料夾
- JSP , JavaBeans, Servlet 用java 編譯工具ant 編譯並封裝成war 檔



七、更多功能

- 網站重爬
- 鑲入網頁語法
- 快速簡易移除程式
- 運算時間統計
- 教學與說明網站
- 解決叢集內IP重複問題
- 刪除索引庫
- 刪除網頁爬取工作

八、Feature Works

- 自動排程
- AJax
- 更多的protocal (FTP, Samba, ….)
- 更新的專案套件（Nutch, Hadoop, …）
- 錯誤修復
- ….

LIVE DEMO I

安裝Crawlzilla

(1) Master 安裝
(2) Slave 安裝

~~Live~~ Video Demo

- 系統安裝(Demo Video also @ YouTube)

Future Work

- 自動排程
- AJAX
- 更多的通訊協定 (FTP, Samba, ….)
- 更新的專案套件(Nutch, Hadoop, …)
- 錯誤修復
- ……

Reference

- J. Dean and S. Ghemawat, MapReduce: Simplified Data Processing on Large Clusters, In Proceedings of the 6th Conference on Symposium on Opearting Systems Design & Implementation - Volume 6, San Francisco, CA, December 06 - 08, 2004.
- S. Ghemawat, H. Gobioff and S. T. Leung, The Google File System, 19th ACM Symposium on Operating Systems Principles, Lake George, NY, October, 2003.
- The Apache Software Foundation, Nutch, available at: http://nutch.apache.org/ , accessed 5 June 2010.
- The Apache Software Foundation, Hadoop, available at: http://hadoop.apache.org/ , accessed 5 June 2010.
- The Apache Software Foundation, Lucene, available at: http://lucene.apache.org/ , accessed 5 June 2010.
- Crawlzilla @ Google Code Project Hosting, available at: http://code.google.com/p/crawlzilla/, accessed 15 Sep 2010.

你也可以擁有自己的搜尋引擎 !!! Start from Here!

- Crawlzilla @ Google Code Project Hosting (中文說明頁)
  - http://code.google.com/p/crawlzilla/
- Crawlzilla @ Source Forge (Toturial in English)
  - http://sourceforge.net/p/crawlzilla/home/
- Crawlzilla User Group @ Google
  - http://groups.google.com/group/crawlzilla-user
- NCHC Cloud Computing Research Group
  - http://trac.nchc.org.tw/cloud

Q & A

感謝您的參與

附錄

HADOOP

HDFS - 檔案系統

HDFS = Hadoop Distributed File System，是Hadoop用來存放資料的分散式檔案系統，因此若要讓Hadoop運算資料，就要將待運算的資料放到這個空間才可以；同理可知運算後的資料...。

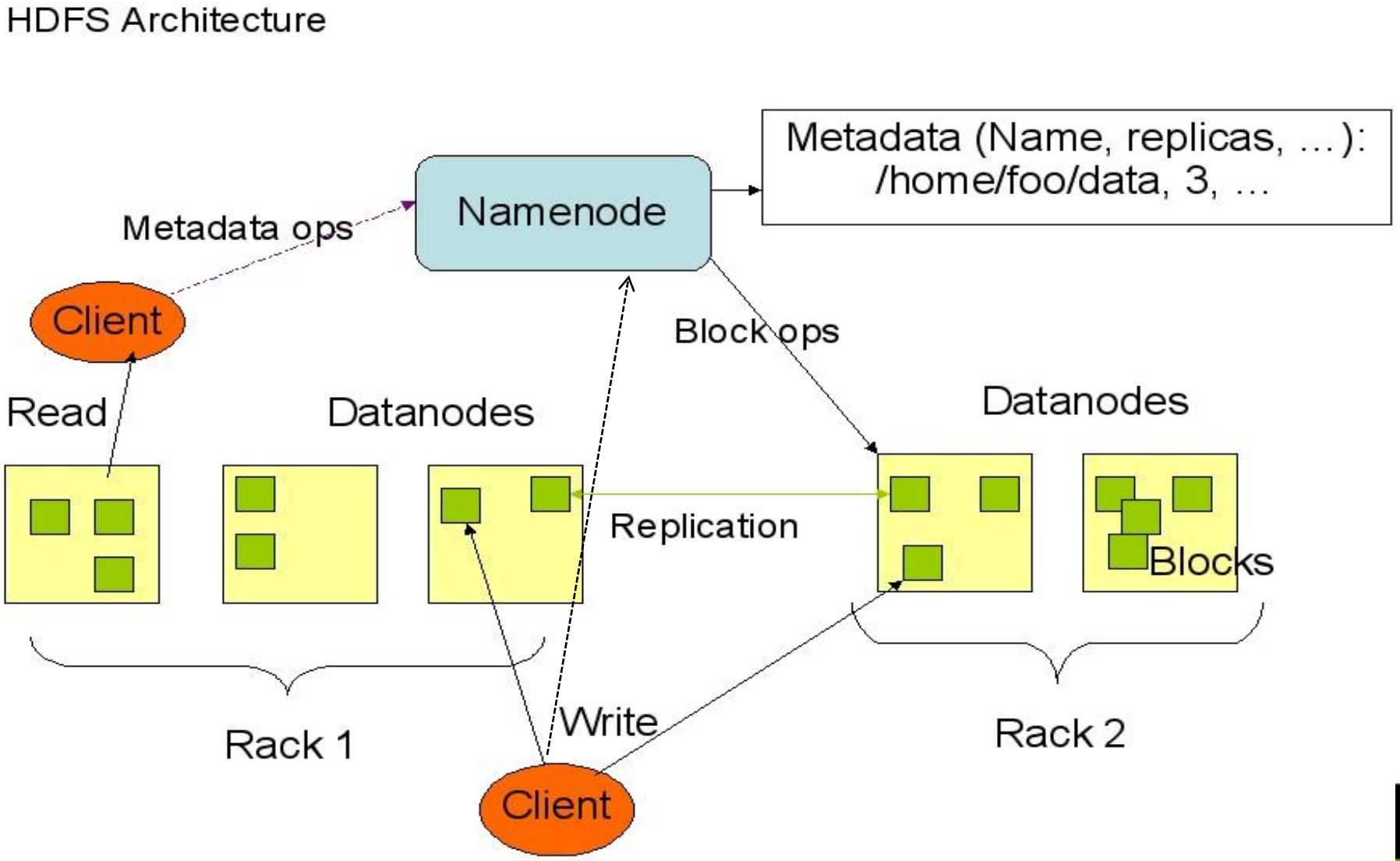
基本假設

- 高單元故障率
  - 便宜的個人電腦商品總是容易故障
- 海量檔案的『客觀』參考數字
  - 就只是幾百萬個檔案而已...
  - 每個大小約100MB或更大; 通常以數GB的檔案最常見
- 這些檔案都只寫一次(write-once)，多數是附加(append)
- 大量的串流讀取需求
- 持續而大量的資料通量(throughput)遠比較短的反應延遲(latency)來得重要

HDFS 設計準則

- 檔案以區塊(block)方式儲存
  - 每個區塊大小遠比多數檔案系統都來得大(預設值為64MB)
- 透過複本機制來提高可靠度
  - 每個區塊至少備分到三台以上的DataNode
- 單一master (NameNode) 來協調存取及屬性資料(metadata)
  - 簡易的集中控管機制
- 沒有資料快取機制(No data caching)
  - 快取對於大資料集與串流讀取沒太大幫助
- 熟悉的介面，但客制化的API
  - 簡化問題；專注於分散式應用

管理資料

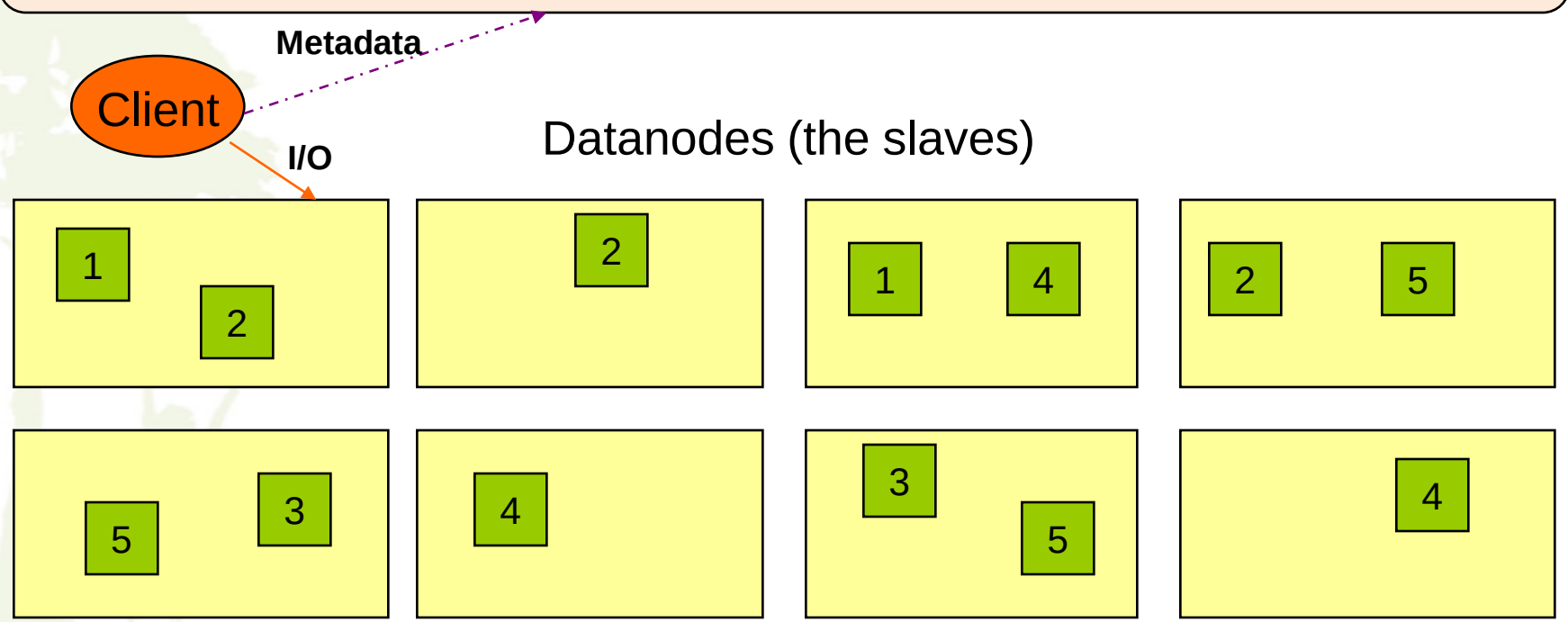

HDFS Architecture
Metadata (Name, replicas, …):
/home/foo/data, 3, …
Namenode
Metadata ops
Client
Block ops
Read
Datanodes
Datanodes
Replication
Blocks
Rack 1
Write
Rack 2
Client

HDFS 運作

Namenode (the master)

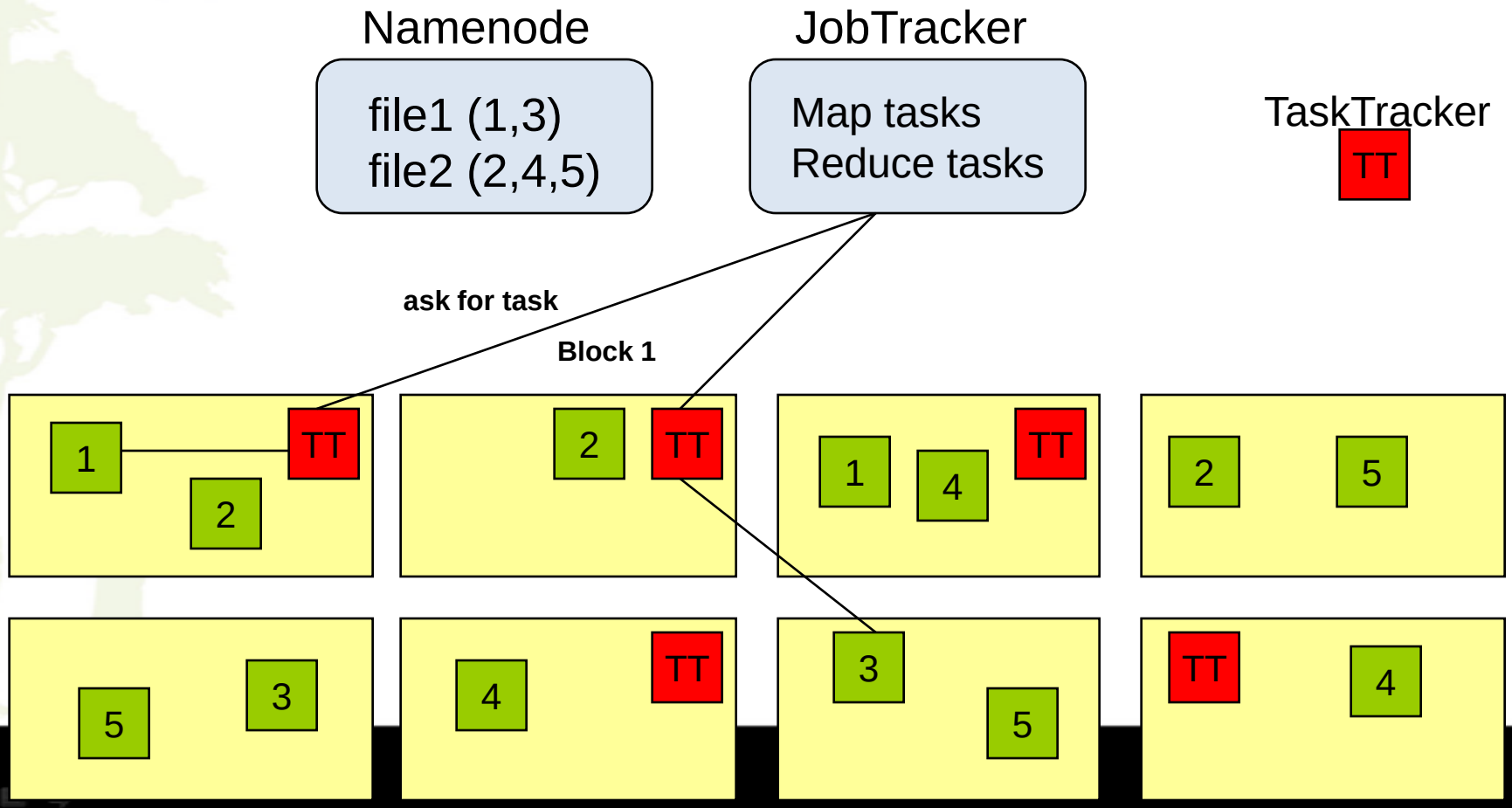
檔案路徑 - 副本數 , 由哪幾個block組成

name:/users/joeYahoo/myFile - copies:2, blocks:{1,3}
name:/users/bobYahoo/someData.gzip, copies:3, blocks:{2,4,5}

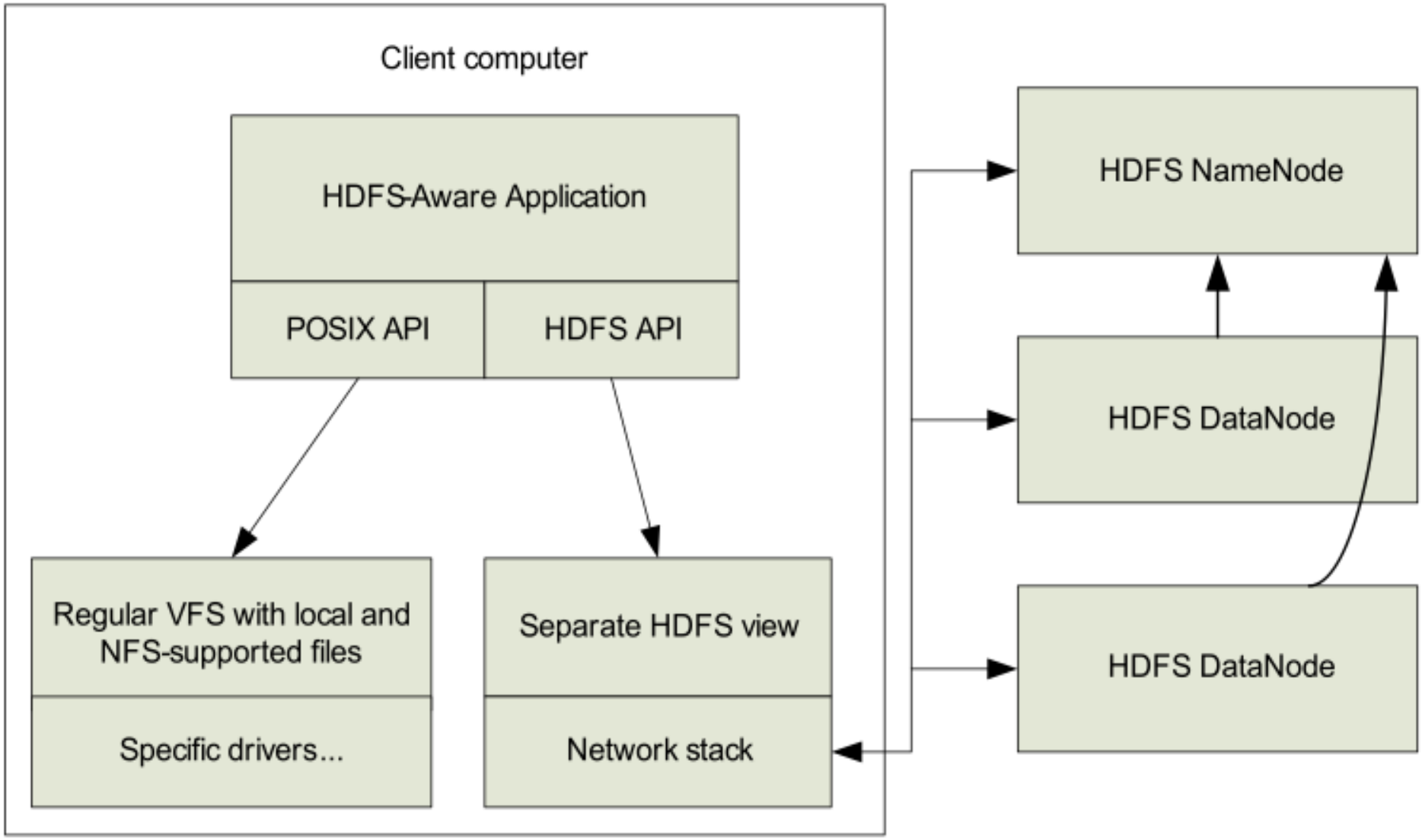


HDFS 運作

- 目的：提高系統的可靠性與讀取的效率
  - 可靠性：節點失效時讀取副本已維持正常運作
  - 讀取效率：分散讀取流量（但增加寫入時效能瓶頸）



HDFS 系統流程圖

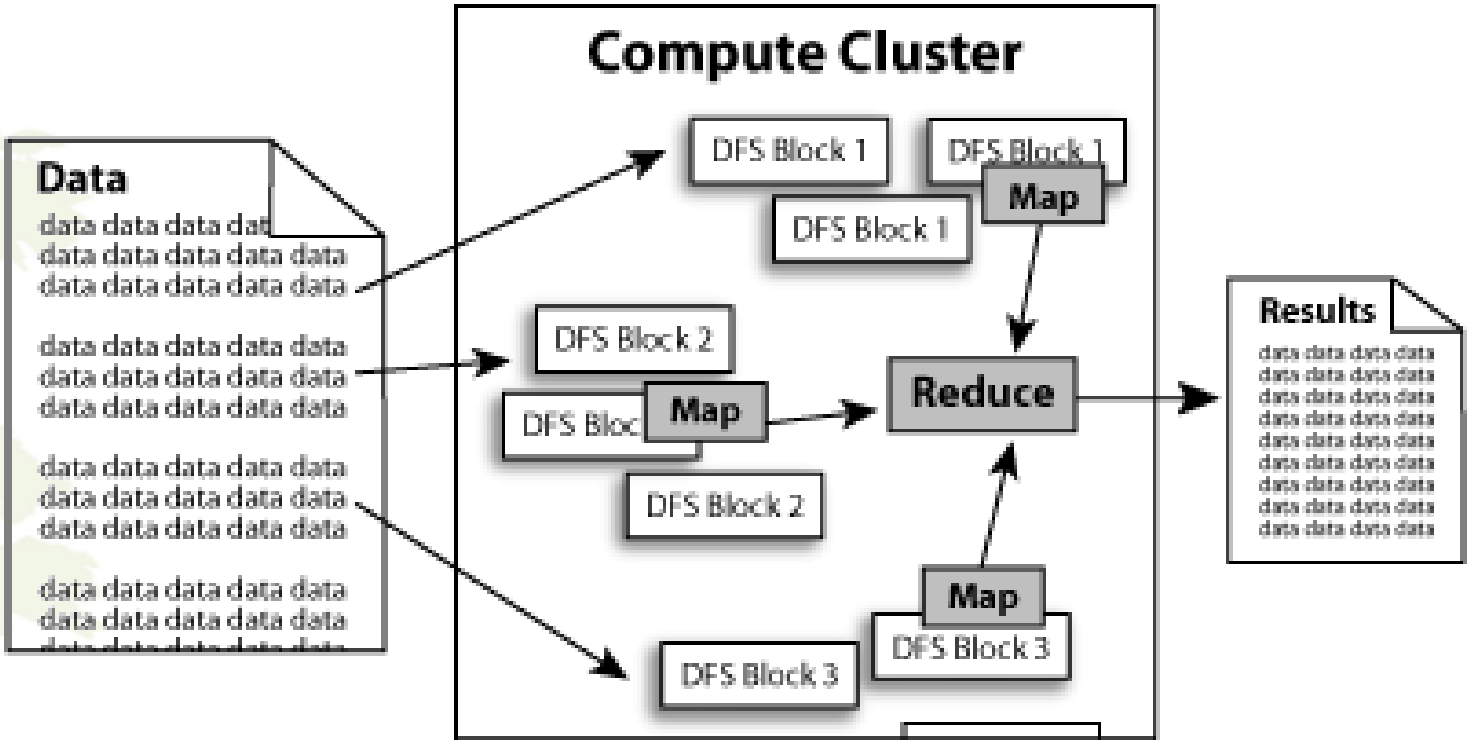

Client computer
HDFS-Aware Application
POSIX API
HDFS API
Regular VFS with local and NFS-supported files
Specific drivers...
Separate HDFS view
Network stack
HDFS NameNode
HDFS DataNode
HDFS DataNode

HADOOP

MapReduce 演算法

Hadoop 的運算方式是透過Map/Reduce
演算法構成，也意謂著，要透過
Hadoop API 撰寫平行分散式程式，則
一定透過 Map / Reduce

Map / Reduce 定義



MapReduce is a software framework to support distributed computing on large data sets on clusters of computers.

演算法

- **Functional Programming : Map Reduce**
  - map(...) :
    - [1,2,3,4] – (*2) -> [2,4,6,8]
  - reduce(...):
    - [1,2,3,4] - (sum) -> 10
  - 對應演算法中的Divide and conquer
  - 將問題分解成很多個小問題之後，再做總和

Map

- **One-to-one Mapper**

```
let map(k, v) =
Emit(k.toUpper(),
v.toUpper())
```

(“Foo”, “other”) → (“FOO”, “OTHER”)
(“key2”, “data”) → (“KEY2”, “DATA”)

- **Explode Mapper**

```
let map(k, v) =
foreach char c in v:
  emit(k, c)
```

(“A”, “cats”) → (“A”, “c”), (“A”, “a”),
(“A”, “t”), (“A”, “s”)

- **Filter Mapper**

```
let map(k, v) =
if (isPrime(v)) then
emit(k, v)
```

(“foo”, 7) → (“foo”, 7)
(“test”, 10) → (nothing)

Reduce

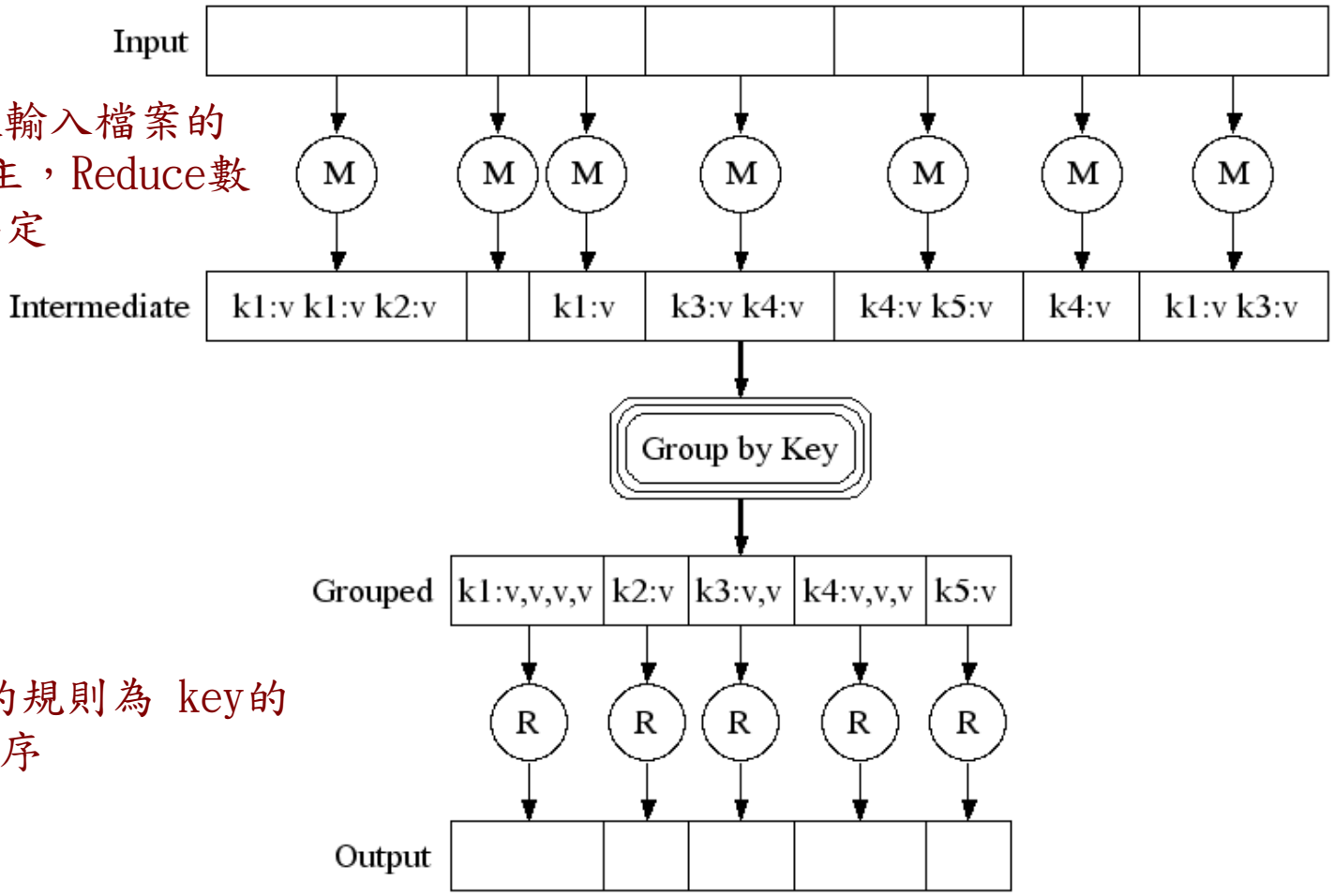
Example: Sum Reducer

```
let reduce(k, vals) =
    sum = 0
    foreach int v in vals:
        sum += v
    emit(k, sum)
```

("A", [42, 100, 312]) → ("A", 454)
("B", [12, 6, -2]) → ("B", 16)

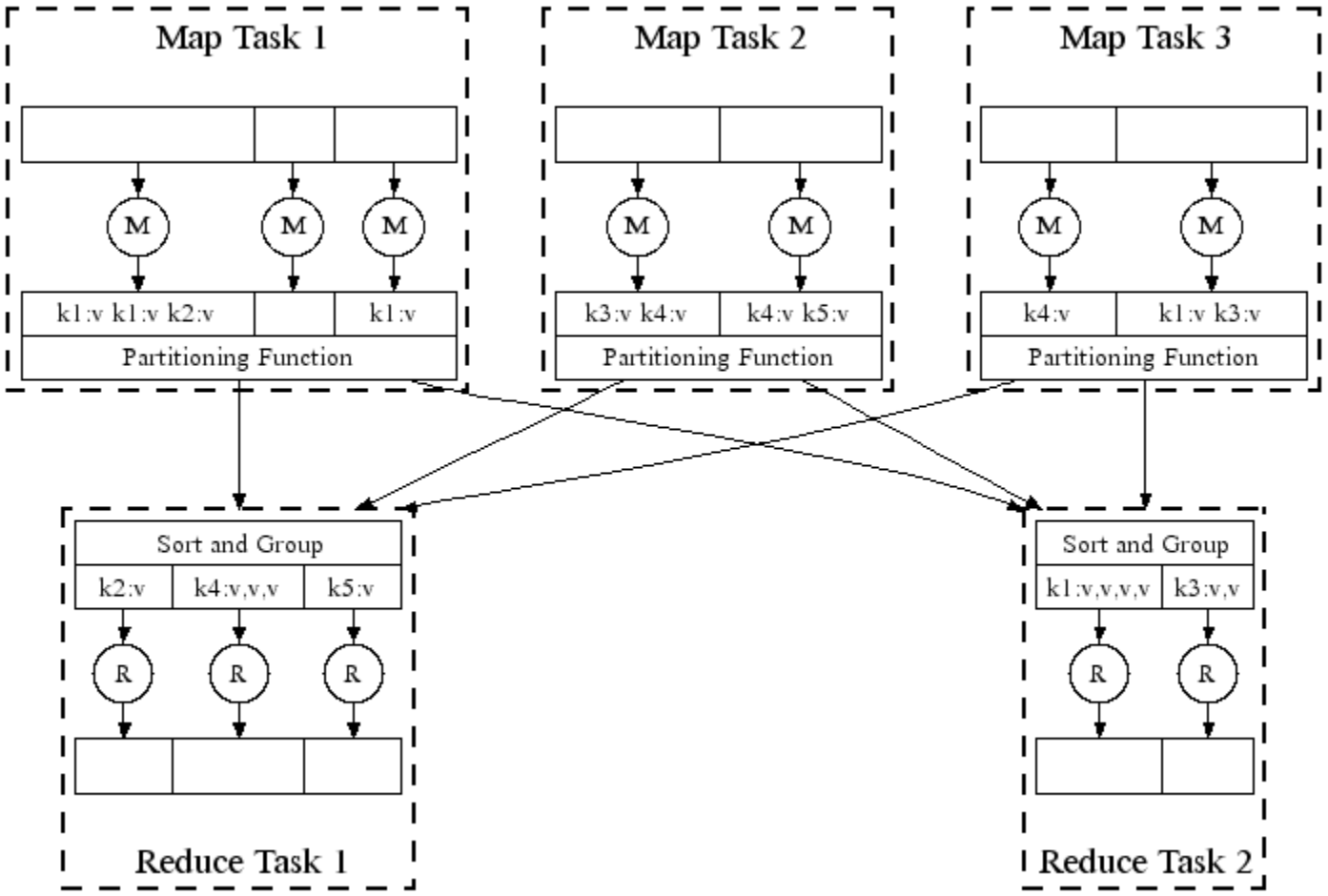
MapReduce 單台

Map 數量依輸入檔案的block數為主，Reduce數量由系統決定

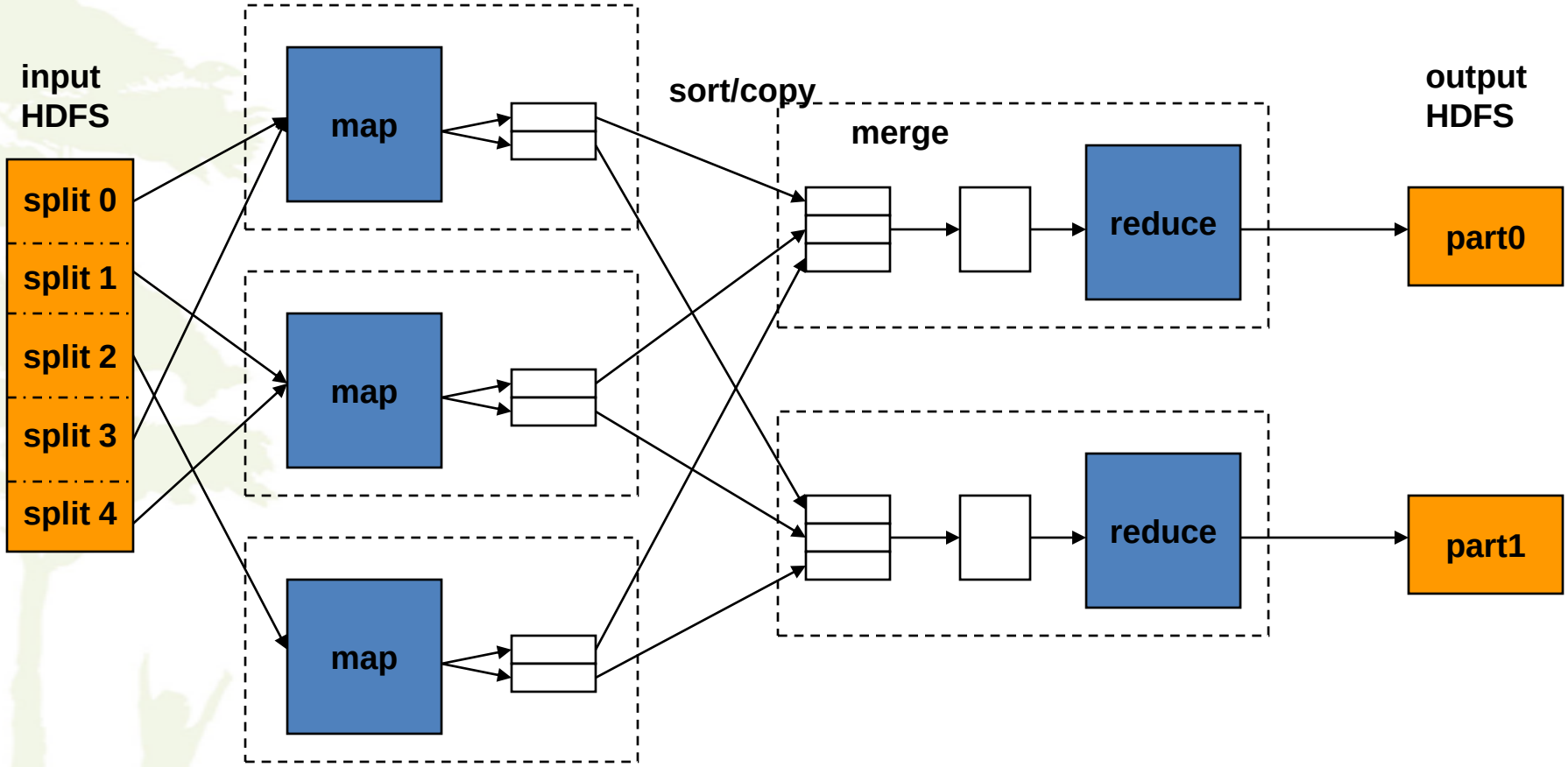


Sort 的規則為 key的字母順序

MapReduce 多台



MapReduce 運作流程



JobTracker跟NameNode取得需要運算的blocks

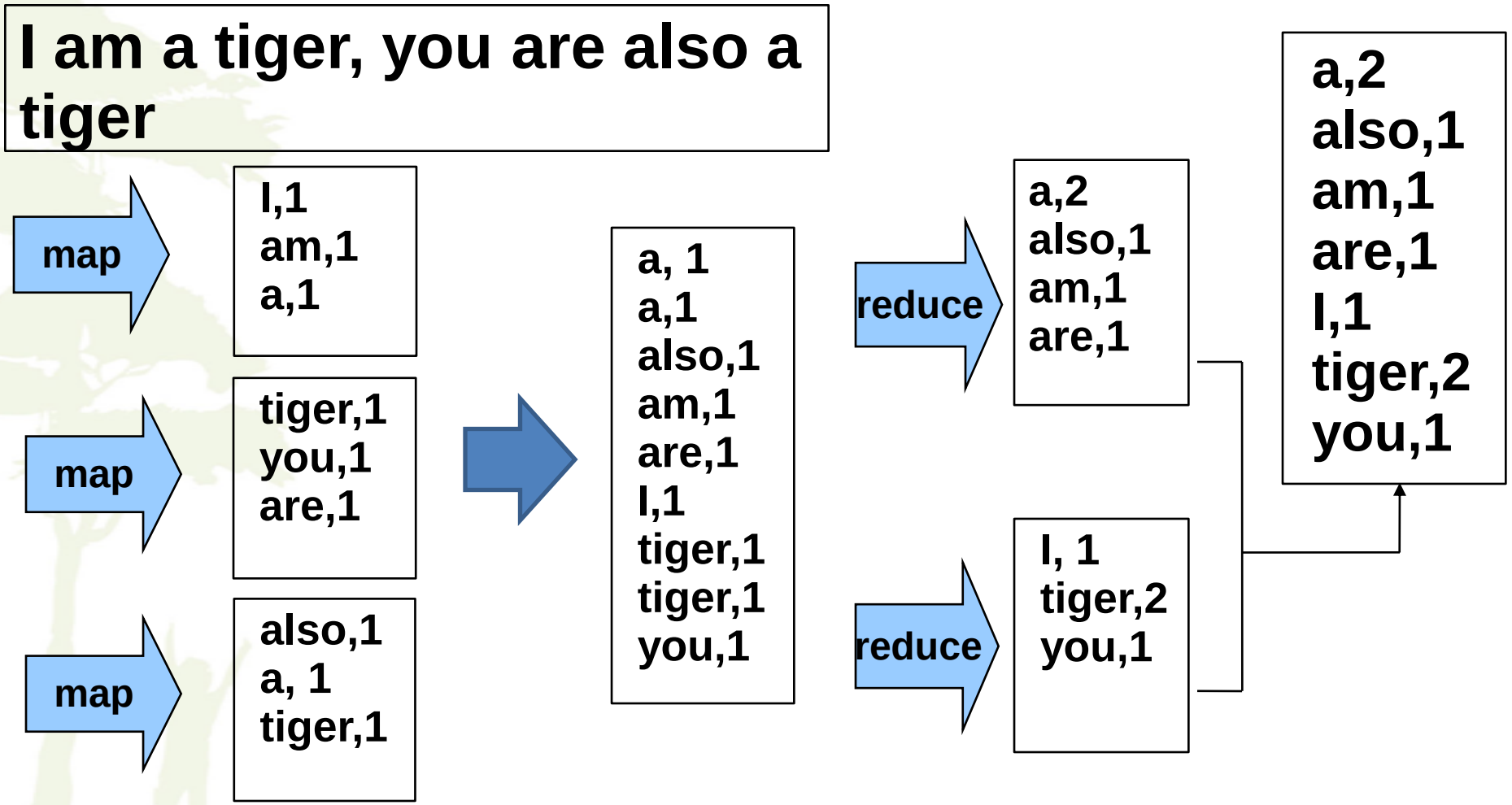
JobTracker選數個TaskTracker來作Map運算，產生些中間檔案

JobTracker將中間檔案整合排序後，複製到需要的TaskTracker去

JobTracker派遣TaskTracker作reduce

reduce完後通知JobTracker與Namenode以產生output

實例範例



JobTracker先選了三個Tracker做map

Map結束後，hadoop進行中間資料的整理與排序

JobTracker再選兩個TaskTracker作reduce