METAHEURÍSTICAS
2018 - 2019

- Tema 1. Introducción a las Metaheurísticas
- Tema 2. Modelos de Búsqueda: Entornos y Trayectorias vs Poblaciones
- Tema 3. Metaheurísticas Basadas en Poblaciones
- Tema 4: Algoritmos Meméticos
- Tema 5. Metaheurísticas Basadas en Trayectorias
- Tema 6. Metaheurísticas Basadas en Adaptación Social
- Tema 7. Aspectos Avanzados en Metaheurísticas
- Tema 8. Metaheurísticas Paralelas

METAHEURÍSTICAS

TEMA 8: METAHEURÍSTICAS EN SISTEMAS DESCENTRALIZADOS (Metaheurísticas Paralelas)

1. INTRODUCCIÓN
2. PROGRAMACIÓN PARALELA/DISTRIBUIDA
3. METAHEURÍSTICAS PARALELAS
4. MÚLTIPLES BÚSQUEDAS INDEPENDIENTES
5. MULTIPLES BÚSQUEDAS COOPERATIVAS
6. MAP-REDUCE Y HADOOP: NUEVA PLATAFORMA PARA EL PROCESAMIENTO DISTRIBUIDO

Bibliografía

- E. Alba (ed.), “Parallel Metaheuristics”, John Wiley & Sons, 2005
- T.G. Crainic, M. Toulouse, Chap. 17: “Parallel Strategies for Metaheuristics.” In F. Glover, G.A. Kochenberber (Eds.) Handbook of Metaheuristics, Kluwer Academic Publisher (2003)
- T.G. Crainic, M. Toulouse, “Parallel Meta-Heuristics” Technical Report (http://www.cirrelt.ca/DocumentsTravail/CIRRELT-2009-22.pdf)

1. Introducción
Objetivos de la paralelización

Las metaheurísticas son, típicamente, algoritmos de cálculo intensivo, es interesante el uso de herramientas de supercomputación

Objetivos de la paralelización

- **Reducir el tiempo de cálculo**
- **Resolver problemas de tamaño mayor en un tiempo dado**
- **Obtener soluciones de mejor calidad sin incrementar el tiempo de cálculo:**
  - **Aumento del número de iteraciones**
  - **Incremento de la diversidad para evitar la convergencia prematura**

2. Programación paralela/distribuida
Conceptos

- Procesamiento paralelo/distribuido significa que varios procesos trabajan simultáneamente en procesadores independientes para resolver un caso concreto de un problema
- El paralelismo surge de una descomposición de la carga de trabajo y de su distribución

2. Programación paralela/distribuida
Rendimiento de un algoritmo paralelo

- La medida básica de rendimiento es la **ganancia** (speed-up):

$$ganancia = \frac{t_{\text{secuencial}}}{t_{\text{paralelo}}}$$

- El límite máximo es el número de procesadores
- Pueden darse ganancias "super-lineales"
- Necesidad de adaptar estos conceptos al ámbito de las metaheurísticas por la dificultad de «realizar el mismo cálculo» (no se obtiene el óptimo).

3. Metaheurísticas paralelas
Metaheurísticas basadas en entornos y en poblaciones

- El proceso de descentralización es diferente según el tipo de metaheurística

- Básicamente, podemos distinguir entre dos tipos:

  1. Algoritmos de Búsqueda basados en Entornos: sólo mantienen una solución actual en cada momento

  2. Algoritmos de Búsqueda basados en Poblaciones: mantienen un conjunto de soluciones en cada momento

4. Múltiples búsquedas independientes

- Esquema simple: varias ejecuciones paralelas e independientes de las metaheurísticas. Lo sorprendente es que es habitualmente da buenos resultados
- La principal desventaja es que no hay intento de intercambio de información entre las ejecuciones independientes, con lo que no se pueden aprovechar del «conocimiento» que van descubriendo sobre el problema
- Se ha aplicado a búsqueda tabú, GRASP, ES, y VNS.

4. Múltiples búsquedas independientes

- No es muy popular para AG porque las al distribuir la población inicial de *n* individuos entre p procesadores surgen poblaciones de tamaño *n*/*p*, que no son tan efectivas como un algoritmo secuencial con una población de tamaño *n*
- Las pequeñas poblaciones aceleran el cálculo pero tienen efectos adversos en la diversidad

5. Múltiples búsquedas cooperativas

- Suponen un paso más en la política de intercambio de información, pues se realiza durante el proceso de búsqueda y no sólo al final. El resultado se traduce en soluciones de mejor calidad que las obtenidas con múltiples ejecuciones paralelas independientes
- El intercambio de información cooperativa especifica cómo interaccionan las metaheurísticas independientes para que el comportamiento emergente en la búsqueda global sea mejor

5. Múltiples búsquedas cooperativas
Ejemplo

- Una estrategia de cooperación podría establecer que un conjunto de metaheurísticas independientes se reinicializasen periódicamente desde la mejor solución actual.

5. Múltiples búsquedas cooperativas
Aspectos importantes

- ¿Qué información intercambiar?
- ¿Entre qué procesos se intercambia la información?
- ¿Cuándo?
- ¿Cómo se intercambia (directo o diferido)?
- ¿Cómo se usa la información importada?

5. Múltiples búsquedas cooperativas
¿Qué información intercambiar?

- La opción más simple es enviar la mejor solución encontrada hasta el momento, pero a lo largo del proceso de búsqueda se tiene mucha más información (p.ej. las memorias en la búsqueda tabú)
- Intercambiar sólo la mejor solución puede ser perjudicial por llevar a una pérdida de diversidad
- La información contextual es importante: información recogida durante la exploración

5. Múltiples búsquedas cooperativas
¿Entre qué procesos se intercambia?

- Intercambio directo entre procesos y definido por la topología de comunicación (estrella, anillo, grid, interconexión total). Comunicación síncrona o asíncrona con buffers.
- Uso de repositorios «centrales» o distribuidos de información (pizarras, *pools*, *data warehouse*). Los procesos envían y cogen información de estos repositorios en lugar de interactuar con otros procesos. Comunicación asíncrona

5. Múltiples búsquedas cooperativas ¿Cuándo y cómo?

- Comunicación síncrona (con esperas) y asíncrona. En cualquier caso, la práctica dice que no debe ser muy frecuente para que los retrasos por comunicaciones no penalicen la ganancia en tiempo y para que no se produzcan convergencias prematuras
- Cooperación síncrona: conseguir información completa del proceso de búsqueda global.
- Cooperación asíncrona: totalmente distribuida, más flexible y permite el desarrollo de un comportamiento emergente más efectivo

5. Múltiples búsquedas cooperativas
Múltiples búsquedas cooperativas basadas en Poblaciones

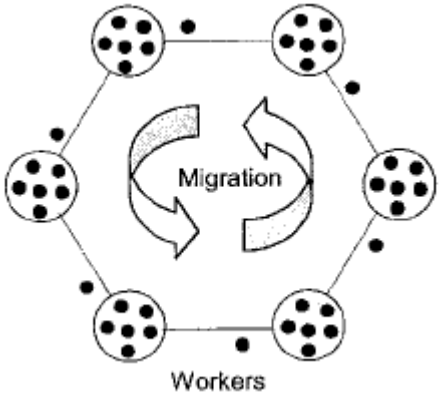
- Una caso especialmente importante por la repercusión habida en artículos de investigación y aplicaciones son las metaheurísticas basadas en poblaciones. Se les puede aplicar cualquiera de los enfoques de paralelización. Pero realmente, resplandecen cuando se trata de búsquedas cooperativas

Múltiples búsquedas cooperativas basadas en poblaciones

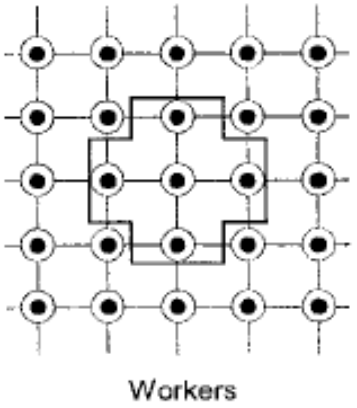
1. Tipo de descentralización
2. Modelos Distribuidos: Modelo de Isla
3. Modelos Celulares o Masivamente Paralelos
4. Relación entre Modelos Distribuidos y Celulares

Múltiples búsquedas cooperativas basadas en poblaciones
1. Tipos de descentralización

- **Distribuidos**
  - Se definen subpoblaciones
  - Comunicación mediante intercambio de individuos



- **Celulares**
  - Sólo hay una población
  - Comunicación mediante vecindad de individuos



Múltiples búsquedas cooperativas basadas en poblaciones

2. Modelos Distribuidos: Modelo de Isla

Fundamento

- En entornos aislados, tales como las islas, se encuentran especies animales que se adaptan más eficazmente a las peculiaridades de su entorno que las correspondientes a superficies de mayor amplitud, esto ha dado lugar a los llamados nichos

Hipótesis

- La competición entre varias subpoblaciones podría proporcionar una búsqueda más efectiva que la evolución de una gran población en la que todos los miembros coexistieran

Propuesta

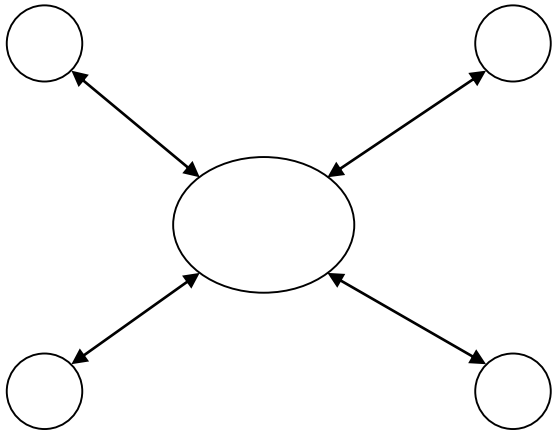
- Modelo Isla: Tener varias poblaciones aisladas que evolucionan en paralelo y periódicamente intercambian por migración sus mejores individuos con las subpoblaciones vecinas

Múltiples búsquedas cooperativas basadas en poblaciones

2. Modelos Distribuidos: Modelo de Isla

Estructuras de Intercomunicación

- **Estrella**: la subpoblación con mayor promedio objetivo se selecciona como maestra y las demás como subordinadas. Todas las subpoblaciones subordinadas envían sus mejores individuos a la maestra, y a su vez, ésta envía también sus mejores individuos a cada una de las subordinadas

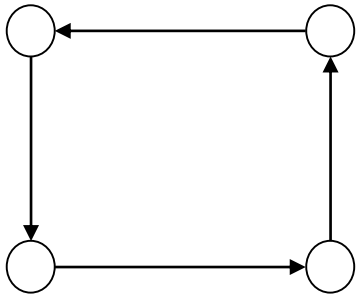


Múltiples búsquedas cooperativas basadas en poblaciones

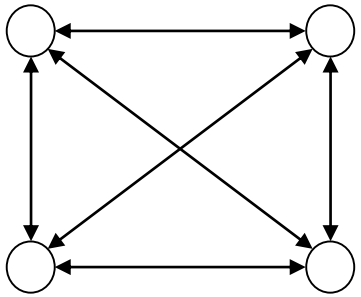
2. Modelos Distribuidos: Modelo de Isla

Estructuras de Intercomunicación

- **Anillo**: Cada subpoblación envía sus mejores individuos a la subpoblación vecina más próxima en un único sentido de flujo



- **Red**: Todas las subpoblaciones envían sus mejores individuos a todas las demás



Múltiples búsquedas cooperativas basadas en poblaciones

3. Modelos Celulares o Masivamente Paralelos

Fundamento

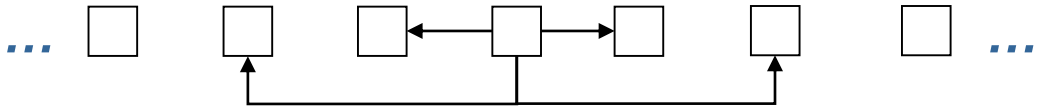
- Se trabaja con una única población y se pretende paralelizar las operaciones que realiza una BP clásica
- Cada individuo es colocado en una celda de un plano cuadriculado. La selección y el cruce se aplican entre individuos vecinos sobre la cuadrícula de acuerdo a una estructura de vecinos preestablecida
- Función de evaluación: Cada procesador elemental debe tener acceso sólo a aquellos individuos para los que calculará su función de evaluación
- Cruce: Cada procesador elemental que cree un nuevo individuo debe tener acceso a todos los otros individuos puesto que cada uno de ellos se puede seleccionar como padre
- Mutación: Cada procesador elemental necesita sólo los individuos con los que trate

Múltiples búsquedas cooperativas basadas en Poblaciones

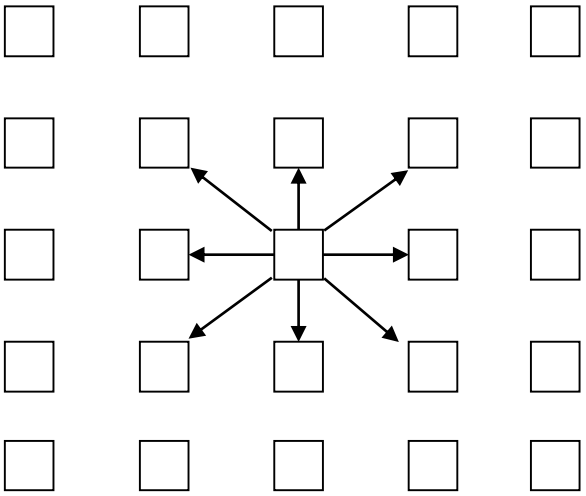
3. Modelos Celulares o Masivamente Paralelos

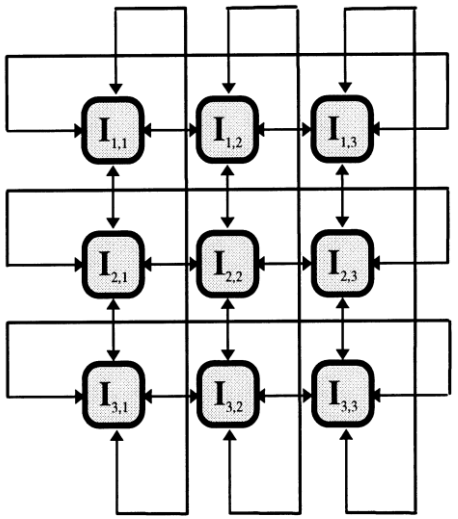
Estructuras de Intercomunicación

- Lista Circular



- Matriz bidimensional

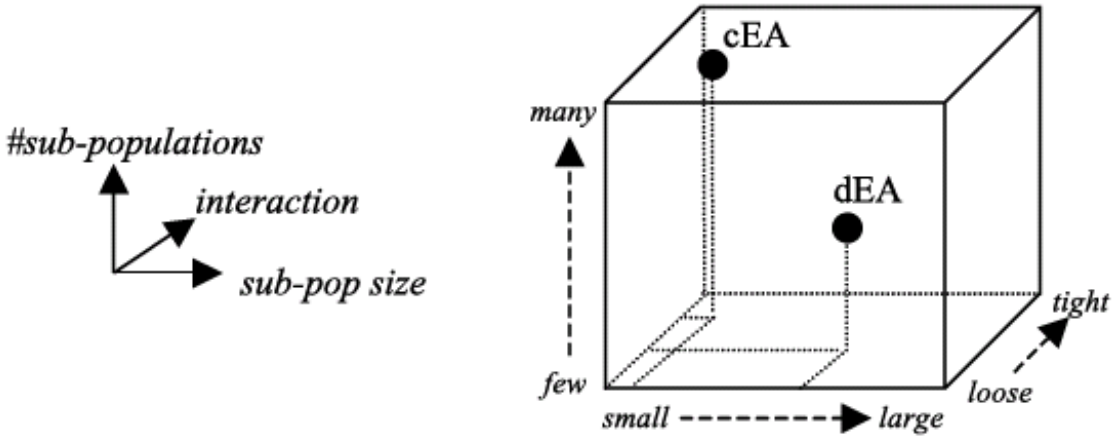




Múltiples búsquedas cooperativas basadas en Poblaciones

3. Relación entre Modelos Distribuidos y Celulares

- dEA = Modelos Distribuidos
- cEA = Modelos Celulares



6. MapReduce y Hadoop: Procesamiento distribuido en la nube

- El modelo de programación paralela de datos de MapReduce oculta la complejidad de la distribución y tolerancia a fallos. Gestión de comunicación entre maquina.

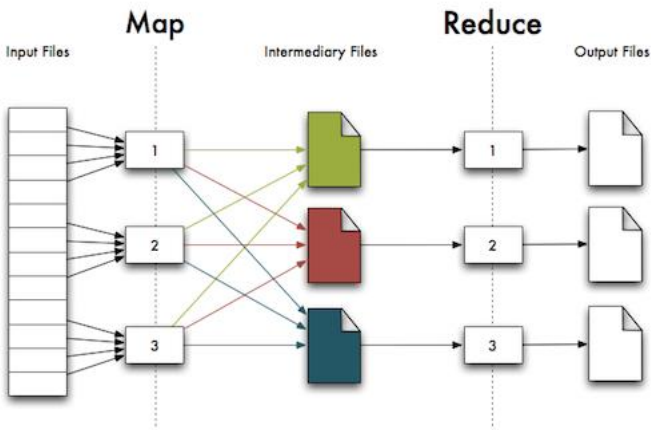
- **Completamente transparente para el programador /analista/usuario**

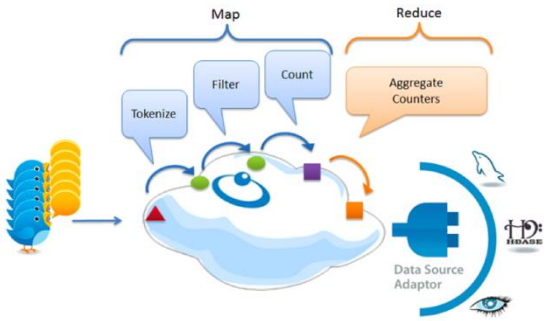
- Key philosophy:
  - *Make it scale (se olvidan los problemas de hardware)*
  - *Make it cheap (se ahorran costes en hardware, programación y administración)*

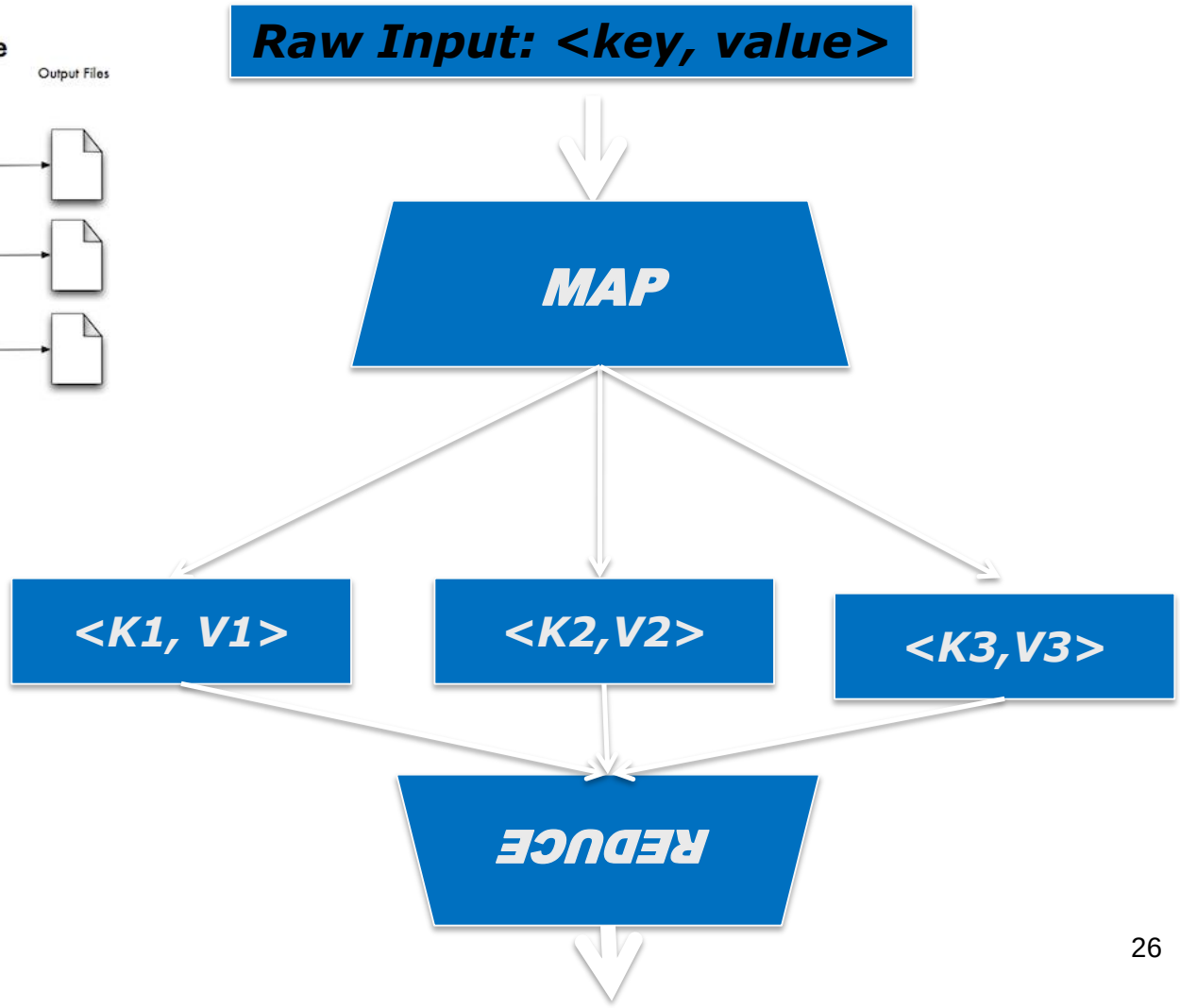
- MapReduce no es adecuado para todos los problemas, pero cuando funciona, puede ahorrar mucho tiempo

6. MapReduce y Hadoop

Programming Framework







6. MapReduce y Hadoop: ¿Por qué Big Data?

- **Problema:** Escalabilidad de grandes cantidades de datos
- **Ejemplo:**
  - Exploración 100 TB en 1 nodo @ 50 MB/sec = 23 días
  - Exploración en un clúster de 1000 nodos = 33 minutos
- **Solución ➔ Divide-Y-Vencerás**

Una sola máquina no puede gestionar grandes volúmenes de datos de manera eficiente

6. MapReduce y Hadoop: ¿Por qué Big Data?

- **Problema:** Escalabilidad de grandes cantidades de datos
- **Ejemplo:**
  - Exploración 100 TB en 1 nodo @ 50 MB/sec = 23 días
  - Exploración en un clúster de 1000 nodos = 33 minutos
- **Solución ➔ Divide-Y-Vencerás**

¿Cómo podemos procesar

1000 TB or 10000 TB?

6. MapReduce y Hadoop: ¿Por qué Big Data?

- Escalabilidad de grandes cantidades de datos
  - Exploración 100 TB en 1 nodo @ 50 MB/sec = 23 días
  - Exploración en un clúster de 1000 nodos = 33 minutos

Solución → Divide-Y-Vencerás

¿Qué ocurre cuando el tamaño de los datos aumenta y los requerimientos de tiempo se mantiene?

Hace unos años: *Había que aumentar los recursos de hardware (número de nodos). Esto tiene limitaciones de espacio, costes, …*

Google 2004: *Paradigma* ***MapReduce***

6. MapReduce y Hadoop: ¿Por qué Big Data?

- Escalabilidad de grandes cantidades de datos
  - Exploración 100 TB en 1 nodo @ 50 MB/sec = 23 días
  - Exploración en un clúster de 1000 nodos = 33 minutos

Solución → Divide-Y-Vencerás

MapReduce

- *Modelo de programación de datos paralela*
- *Concepto simple, elegante, extensible para múltiples aplicaciones*

- ***Creado por Google (2004)***
  - *Procesa 20 PB de datos por día (2004)*
- ***Popularizado por el proyecto de codigo abierto Hadoop***
  - *Usado por Yahoo!, Facebook, Amazon, …*

6. MapReduce y Hadoop: Procesamiento distribuido en la nube

- **Resumen:**
  - Modelo de programación de datos paralela
  - Una implementación asociada paralela y distribuida
- **Iniciado por Google**
  - Procesa 20 PB de datos por día
- **Popularizado por el proyecto de codigo abierto Hadoop**
  - Usado por Yahoo!, Facebook, Amazon, y la lista sigue creciendo…

6. MapReduce y Hadoop: Procesamiento distribuido en la nube

Experiencia y Contribución

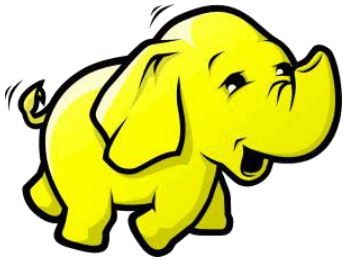
- Funciona sobre grandes clusters:
  - 1000s a 10,000s máquinas
- Procesa muchos terabytes
- Mul fácil de utilizar puesto que la complejidad está oculta a los usuarios

Simple & Powerful Programming Paradigm For Large-scale Data Analysis

Run-time System For Large-scale Parallelism & Distribution

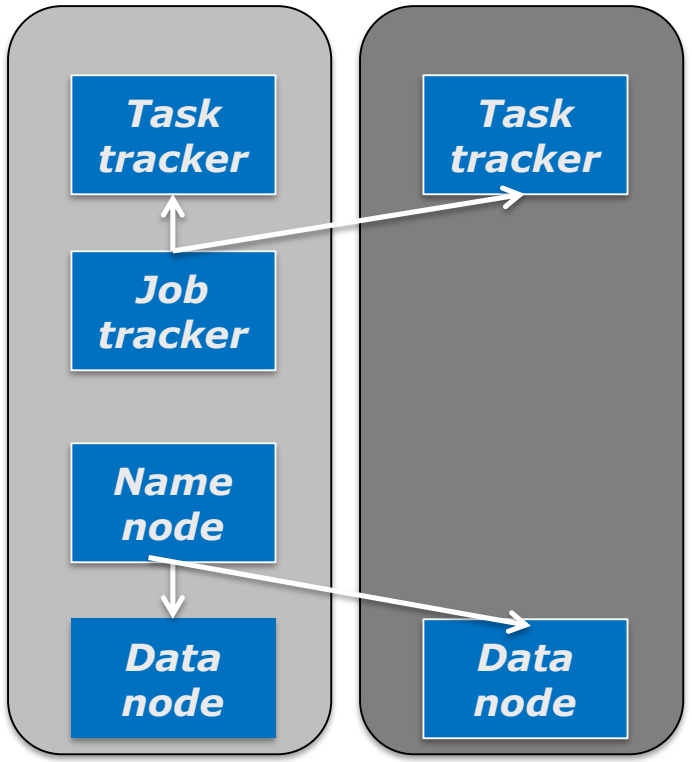
6. MapReduce y Hadoop: Procesamiento distribuido en la nube

Hadoop Distribued File System (HDFS) ***es un sistema de archivos distribuido, escalable y portátil escrito en Java para el framework Hadoop.***



Map Reduce Layer

HDFS Layer



Creado por Doug Cutting (chairman of board of directors of the Apache Software Foundation, 2010)

http://hadoop.apache.org/

6. MapReduce y Hadoop: Procesamiento distribuido en la nube

July 2008 - Hadoop Wins Terabyte Sort Benchmark

One of Yahoo's Hadoop clusters sorted 1 terabyte of data in 209 seconds, which beat the previous record of 297 seconds in the annual general purpose (Daytona) terabyte short bechmark. This is the first time that either a Java or an open source program has won.

What Is Apache Hadoop?

The Apache™ Hadoop® project develops open-source software for reliable, scalable, distributed computing.

http://hadoop.apache.org/

6. MapReduce y Hadoop: Procesamiento distribuido en la nube

El proyecto incluye los módulos:

Hadoop Common: Las utilidades comunes que apoyan los otros módulos de Hadoop.
Hadoop Distribites File System (HDFS): El sistema de ficheros que proporcina el acceso
Hadoop YARN: Un marco para el manejo de recursos de programación y grupo de trabajo.
Hadoop MapReduce: Un sistema de basado en YARN o para el procesamiento en paralelo de grandes conjuntos de datos.

Otros proyectos en Apache Hadoop incluyen:

Avro: Un sistema de serialización de datos.
Cassandra: Una base de datos escalable multi-master si puntos indiviaules e fallo
Chukwa: Un sistema de recogida de datos para la gestión de grandes sistemas distribuidos.
Hbase: Una base de datos distribuida, escalable que soporta estructurado de almacenamiento de datos para tablas de gran tamaño.
Hive: Un almacén de datos que proporciona el Resumen de datos para tablas de gran tamaño.
Mahout: Aprendizaje automático escalable y biblioteca de minería de datos.
Pig: Lenguaje para la ejecución de alto nivel de flujo de datos para computación paralela.
ZooKeeper: Un servicio de coordinación de alto rendimiento para aplicaciones distribuidas.

http://hadoop.apache.org/

6. MapReduce y Hadoop: Procesamiento distribuido en la nube

¿Cómo puedo instalar Hadoop?

*Distribución que ofrece **Cloudera** para Hadoop.*

cloudera
Ask Bigger Questions

http://www.cloudera.com/content/cloudera/en/why-cloudera/hadoop-and-big-data.html

¿Qué es Cloudera?

Cloudera es la primera distribución Apache Hadoop comercial y no-comercial.

- *Incluye el pack Hadoop integrado y completo donde todas las versiones de componentes y dependencias son gestionadas por el usuario.*
- *Trabaja con un gran abanico de sistemas operativos, hardware, bases de datos y herramientas BI (Business Intelligence)*
- *Es fiable y estable: Probado a fondo por Control de Calidad de Cloudera. Demostrado a escala de decenas de entornos empresariales.*
- *Código abierto: Sin restricciones de propiedad pues sólo incorpora componentes de la comunidad de código abierto. Descarga gratuita.*

Conclusiones

- **El procesamiento distribuido/paralelo aporta diferentes ventajas al uso de las metaheurísticas.**
  - **Reducir el tiempo de cálculo**
  - **Resolver problemas de tamaño mayor en un tiempo dado**
  - **Obtener soluciones de mejor calidad sin incrementar el tiempo de cálculo:**
    - **Aumento del número de iteraciones**
    - **Incremento de la diversidad para evitar la convergencia prematura**