MapReduce

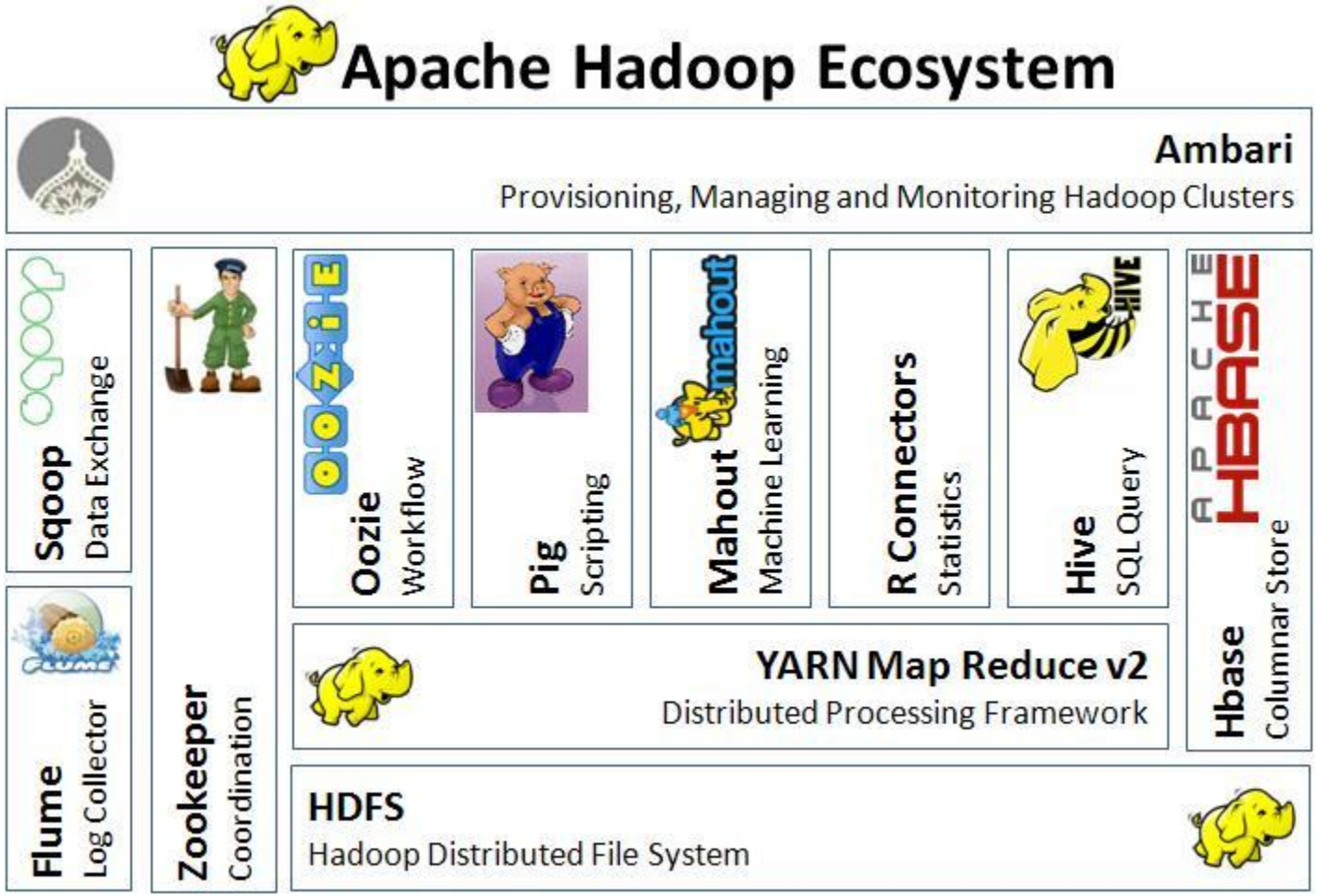
Repaso Hadoop

Que es Hadoop?

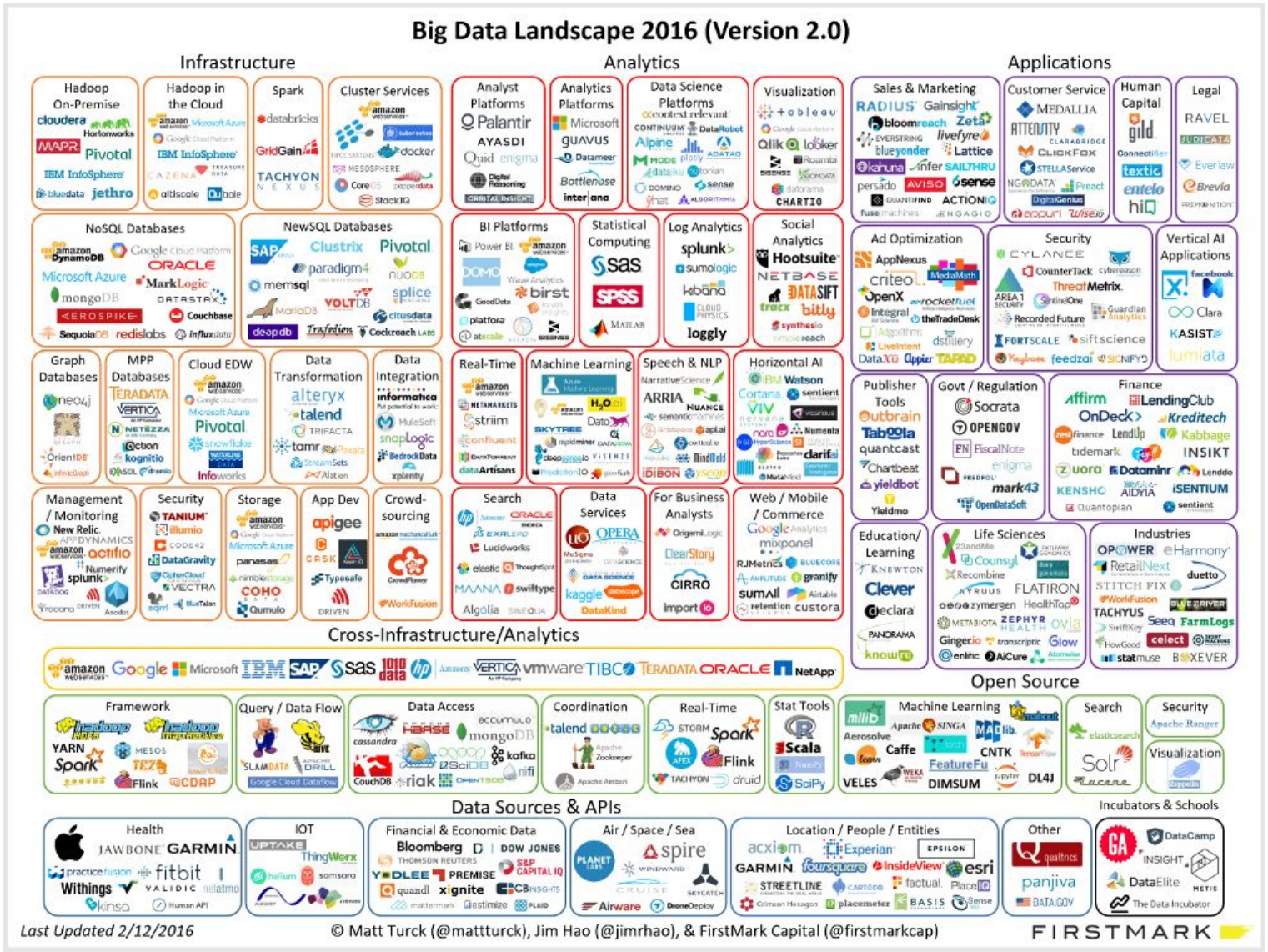
Repaso Hadoop

Que es Hadoop?

Repaso Hadoop



Repaso Hadoop



UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

HDFS

De las siglas **Hadoop Distributed File System**

- Sistema de archivos distribuido
- Pensado para trabajar con grandes volúmenes de información (PB)
- Tolerante a fallas
- Altamente usado para procesamiento de grandes volúmenes de información

Componentes:

HDFS

De las siglas **Hadoop Distributed File System**

- Sistema de archivos distribuido
- Pensado para trabajar con grandes volúmenes de información (PB)
- Tolerante a fallas
- Altamente usado para procesamiento de grandes volúmenes de información

Componentes:

Namenode

- Master del cluster
- Responsable de la metadata de los archivos
- Coordina la salud de los datanodes

Datanode

- Worker del cluster
- Responsable de almacenar la información
- Desconoce el significado de la información que contiene

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

YARN

De las siglas **Yet Another Resource Negotiator**

- Manejador de recursos del cluster
  - Mantiene una visión global de los recursos disponibles (RAM, CPU)
  - Brinda permisos de uso de recursos
  - Controla promesas de uso (**containers**)

Componentes:

YARN

De las siglas **Yet Another Resource Negotiator**

- Manejador de recursos del cluster
  - Mantiene una visión global de los recursos disponibles (RAM, CPU)
  - Brinda permisos de uso de recursos
  - Controla promesas de uso (**containers**)

Componentes:

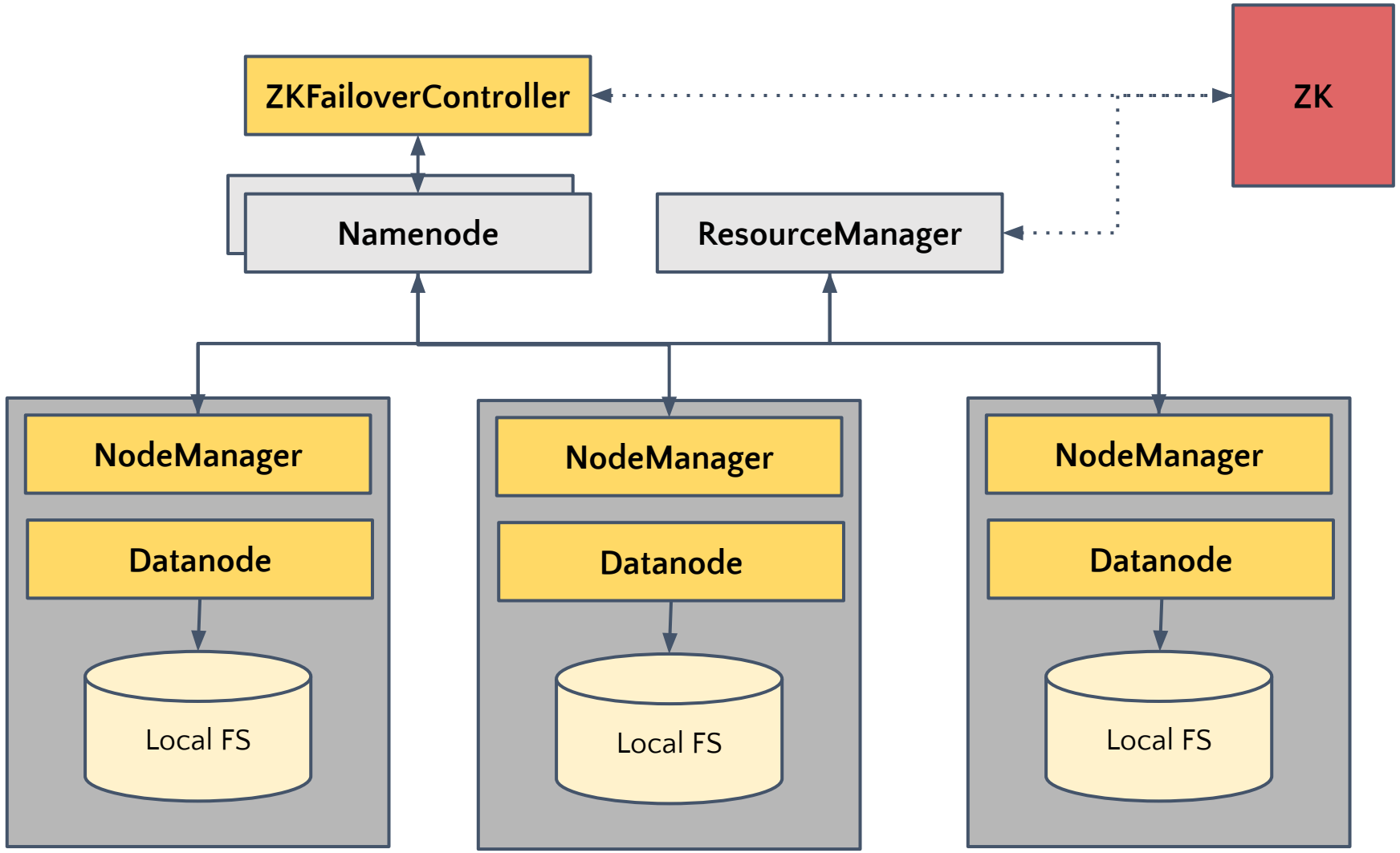
Resource Manager

- Master de YARN
- Responsable de la visión global del cluster
- Punto de contacto para solicitar recursos

Node Manager

- Slave del cluster
- Responsable por informar la disponibilidad de recursos locales
- Responsable de controlar el uso de recursos

Arquitectura



Integraciones

Integraciones:

- MapReduce
- Hive
- Spark
- Flink
- Otros

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Map Reduce

Suponga que usted está en su casa y su principal tarea es la elaboración de alfajores de maicena.

Se puede simplificar el trabajo en 2 etapas:

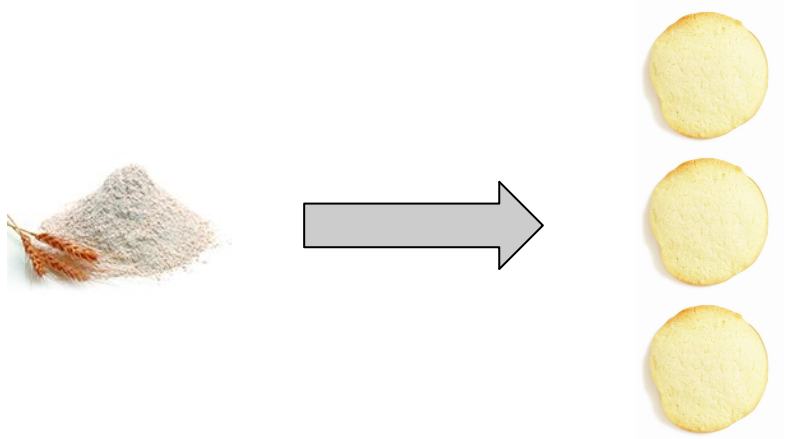
- Elaboración de tapas
- Armado de alfajores

Map Reduce

Suponga que usted está en su casa y su principal tarea es la elaboración de alfajores de maicena.

Se puede simplificar el trabajo en 2 etapas:

- Elaboración de tapas
- Armado de alfajores

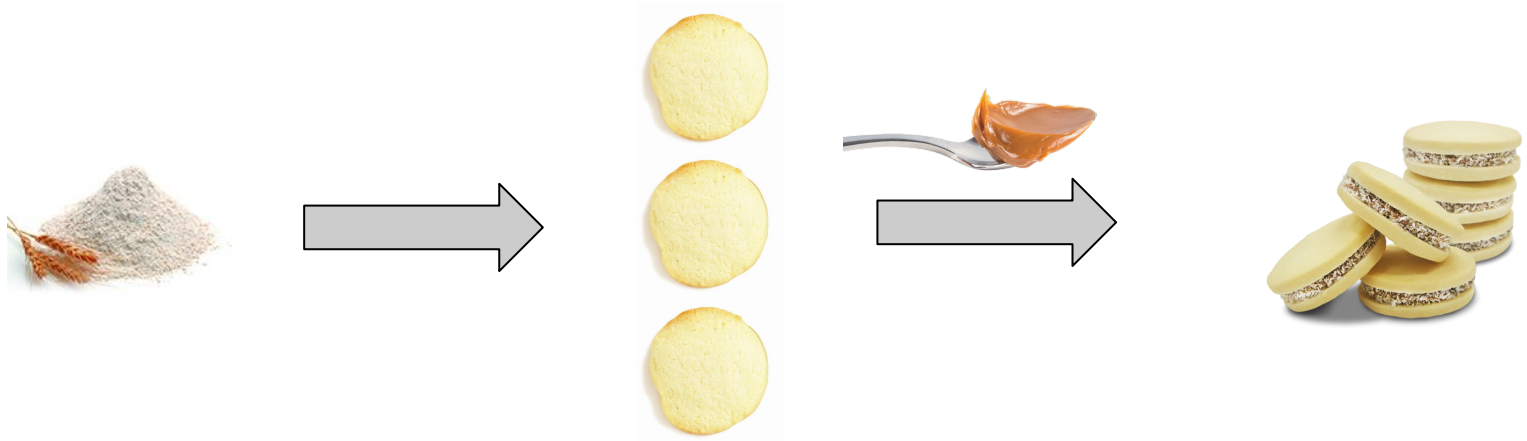


Map Reduce

Suponga que usted está en su casa y su principal tarea es la elaboración de alfajores de maicena.

Se puede simplificar el trabajo en 2 etapas:

- Elaboración de tapas
- Armado de alfajores



Map Reduce

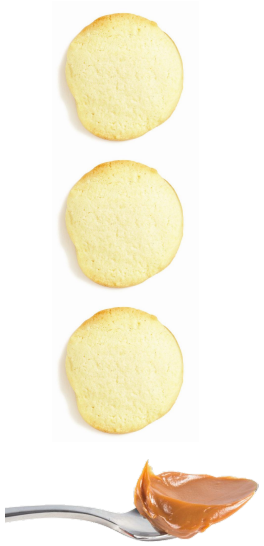
Ahora imaginemos una fábrica de alfajores.

Materia prima

Map Reduce

Ahora imaginemos una fábrica de alfajores.

Materia prima



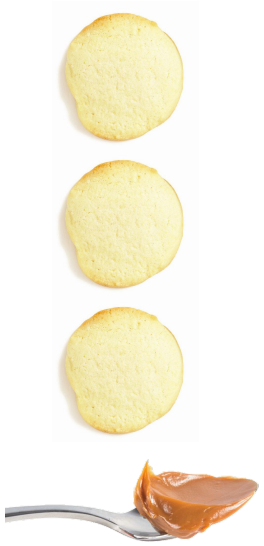
?

Si bien es posible utilizar el mismo método que para hacer alfajores en casa, repetir la técnica en esta circunstancia es receta de fracaso.

Map Reduce

Ahora imaginemos una fábrica de alfajores.

Materia prima



?

Si bien es posible utilizar el mismo método que para hacer alfajores en casa, repetir la técnica en esta circunstancia es receta de fracaso.

MapReduce nos permite hacer exactamente eso.

- Conservar simples primitivas
- Tratar un problema de gran escala de manera simple

Map Reduce

Map reduce es un método de procesamiento distribuído.

- Diseñado en google y abierto al público en 2004 a través del paper **MapReduce: Simplified Data Processing on Large Clusters**

- Derivado del paradigma funcional y compuesto por dos primitivas (Map y Reduce)

- Integrado con hadoop, aunque varias bbdd del ecosistema NoSQL tienen sus propias implementaciones e integraciones

- Permite al usuario programar solamente las primitivas (map/reduce) y olvidarse de las problemáticas de la programación distribuida

  - Sincronización
  - Coordinación
  - Chequeo de salud

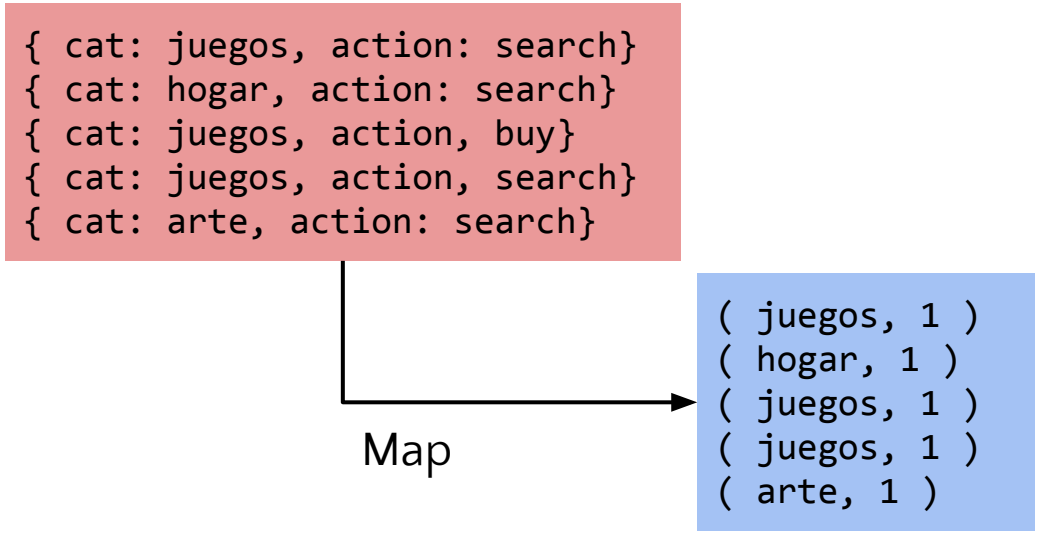
  - Distribución adecuada
  - Tolerancia a fallos
  - Particiones de red

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

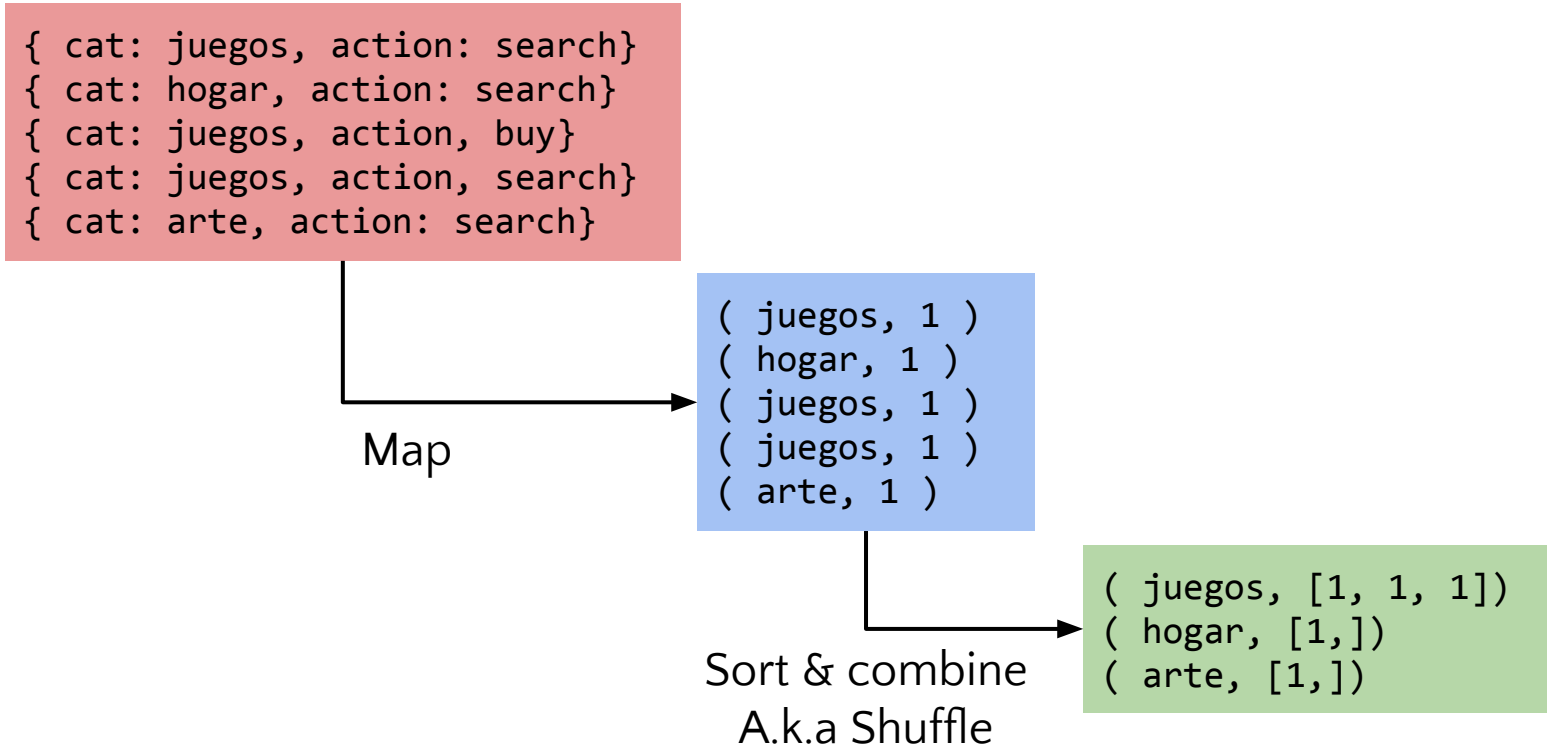
Map Reduce

```
{ cat: juegos, action: search}
{ cat: hogar, action: search}
{ cat: juegos, action, buy}
{ cat: juegos, action, search}
{ cat: arte, action: search}
```

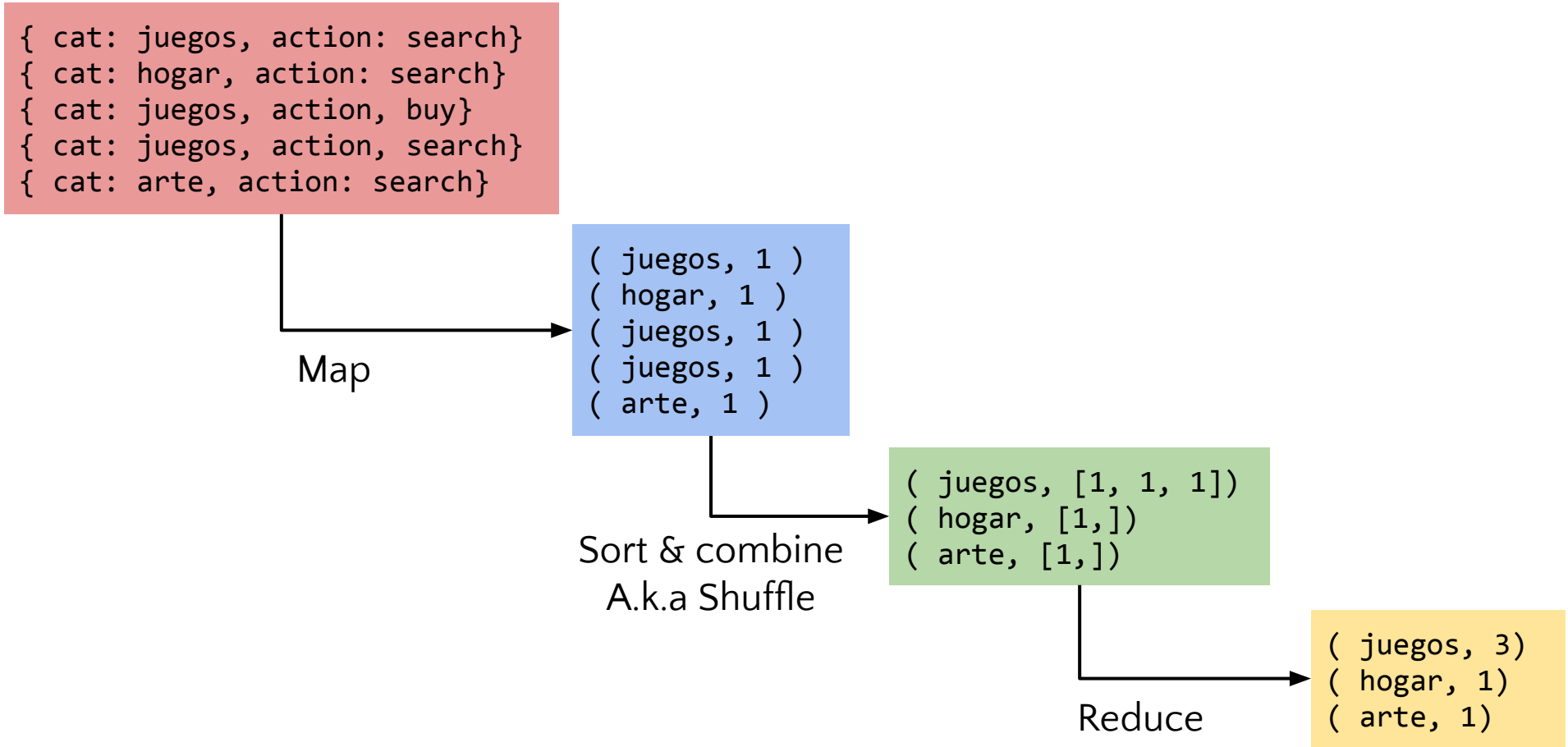
Map Reduce



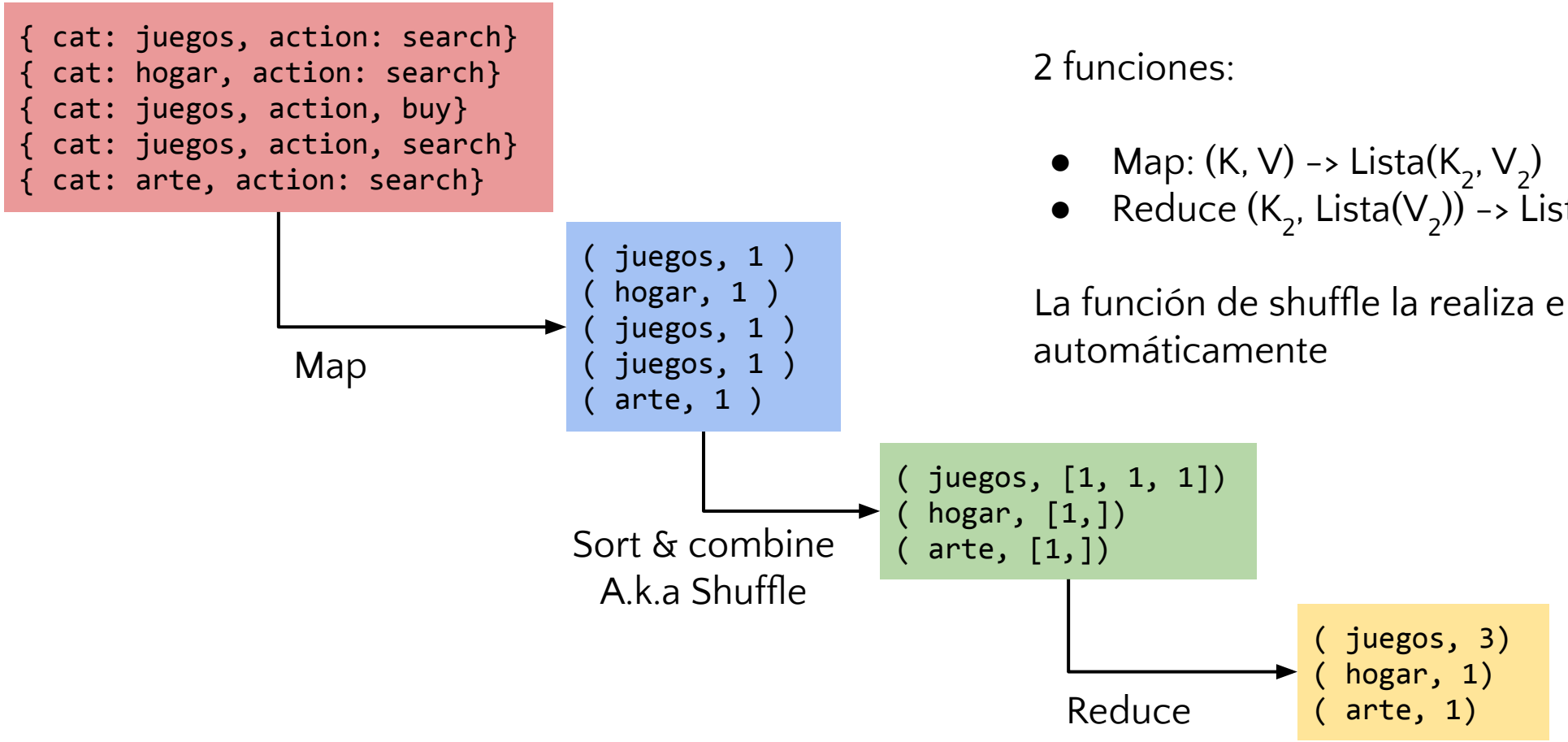
Map Reduce



Map Reduce



Map Reduce



2 funciones:

- Map: (K, V) -> Lista(K_2, V_2)
- Reduce (K_2, Lista(V_2)) -> Lista(K_3, V_3)

La función de shuffle la realiza el motor automáticamente

Map Reduce

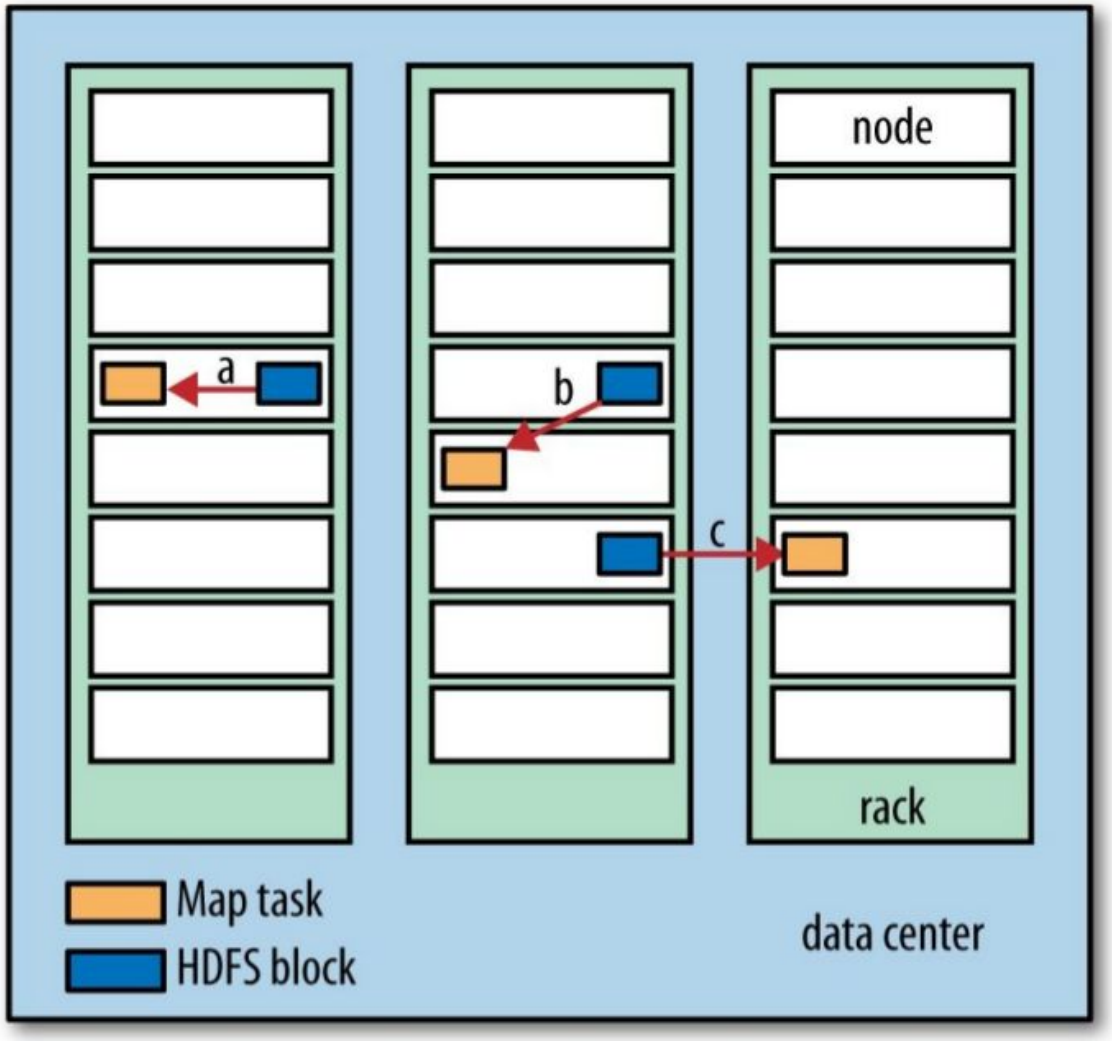
Map Reduce se apoya fuertemente en Hadoop HDFS y YARN para garantizar

- Uso apropiado de recursos (RAM, CPU)
- Planificación de tareas
- **Localidad de datos**

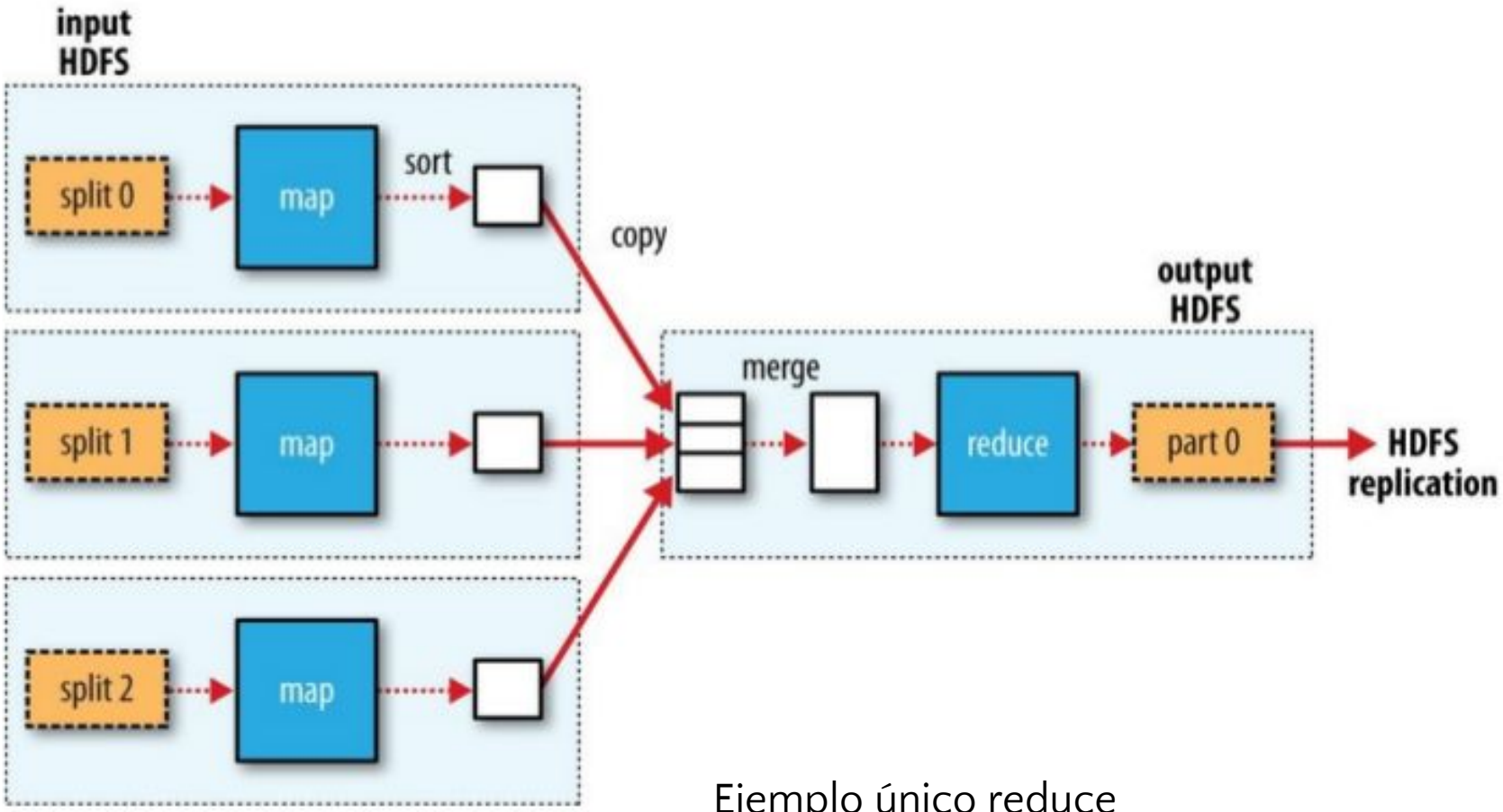
Map Reduce

La entrada al proceso (generalmente HDFS) se encuentra ya particionada y distribuida

- Hadoop MapReduce divide las tareas de map
- Se ejecuta un map por cada "**split**" del archivo en HDFS

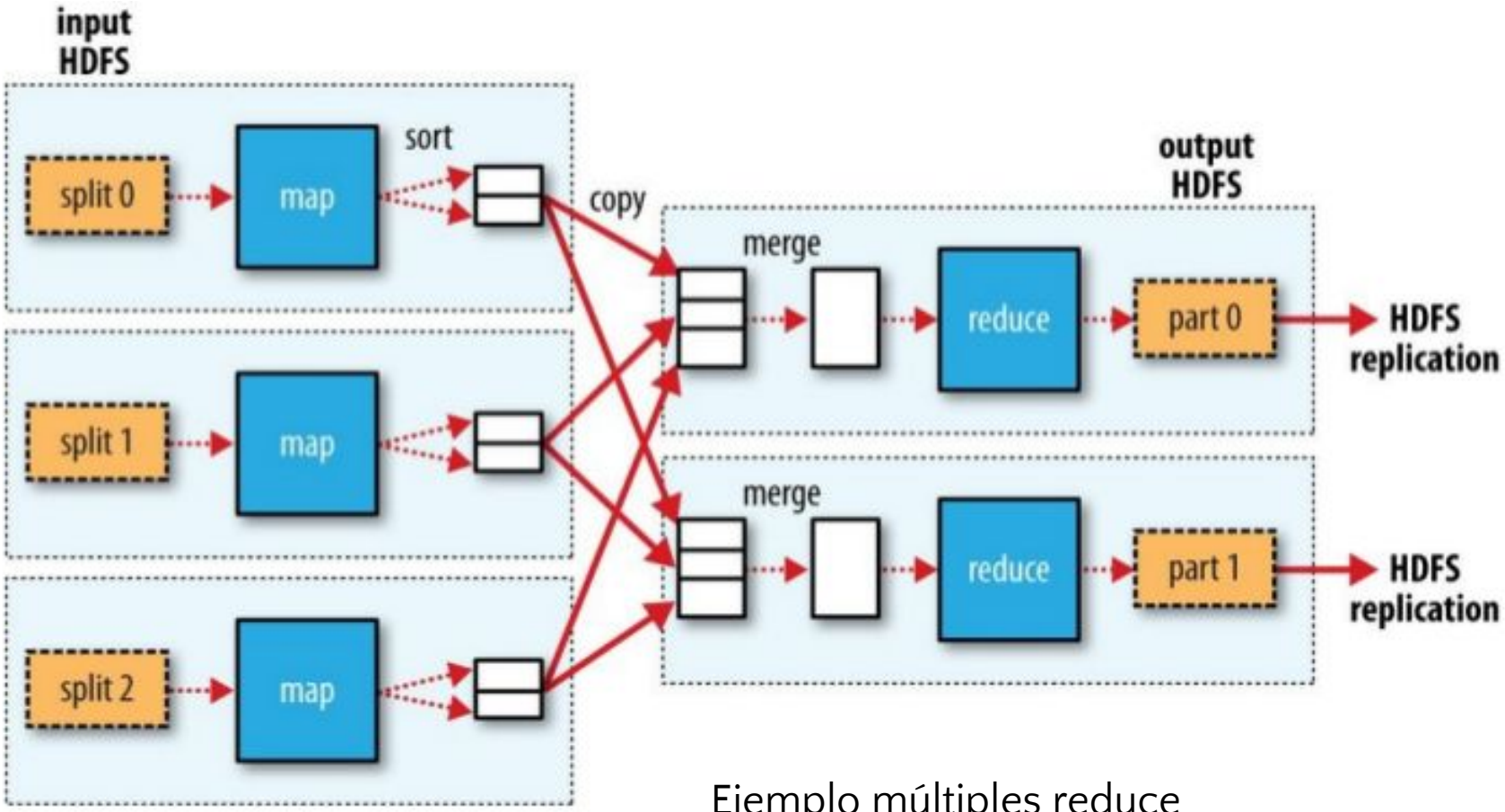


Map Reduce



Ejemplo único reduce

Map Reduce



Ejemplo múltiples reduce

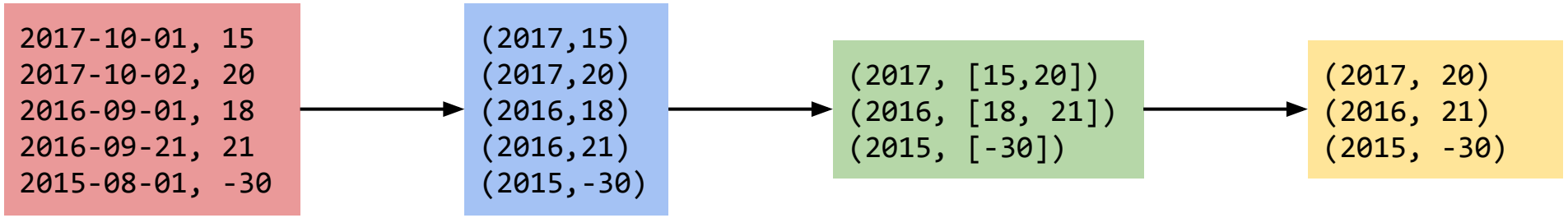
MapReduce (parte II)

Map Reduce

Dado el siguiente dataset: (<temperatura, día>). Consideremos el siguiente ejemplo:

Obtener la máxima temperatura del año a través de MR

Un enfoque válido podría ser:

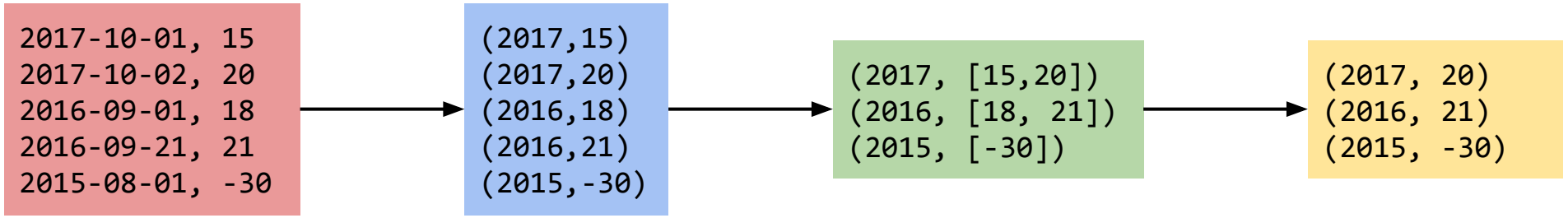


Map Reduce

Dado el siguiente dataset: (<temperatura, día>). Consideremos el siguiente ejemplo:

Obtener la máxima temperatura del año a través de MR

Un enfoque válido podría ser:



Sin embargo...

Map Reduce

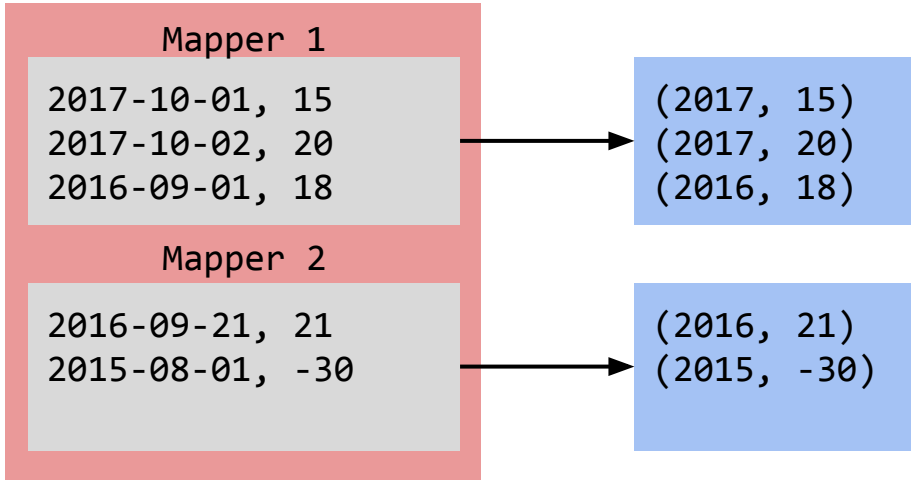
Mapper 1

```
2017-10-01, 15
2017-10-02, 20
2016-09-01, 18
```

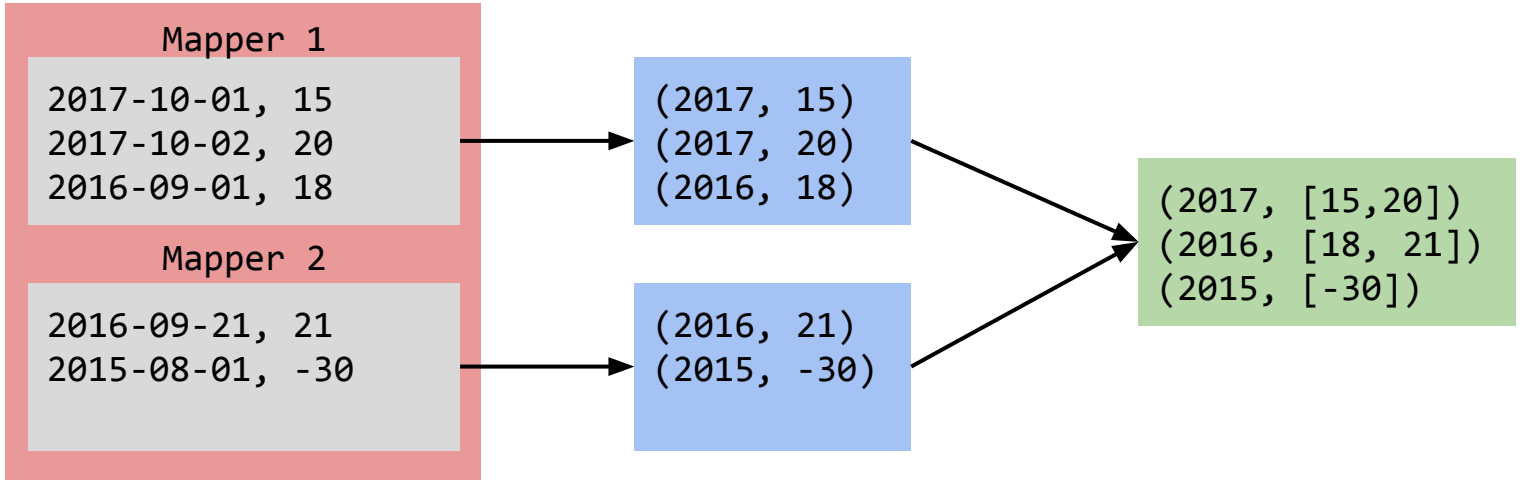
Mapper 2

```
2016-09-21, 21
2015-08-01, -30
```

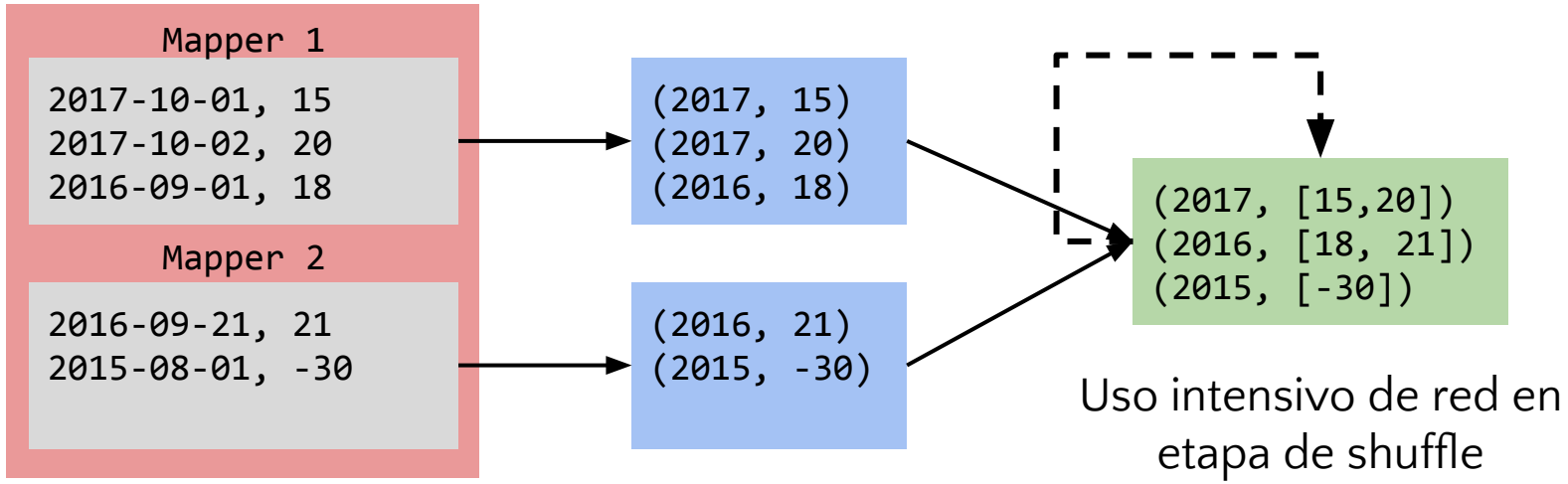
Map Reduce



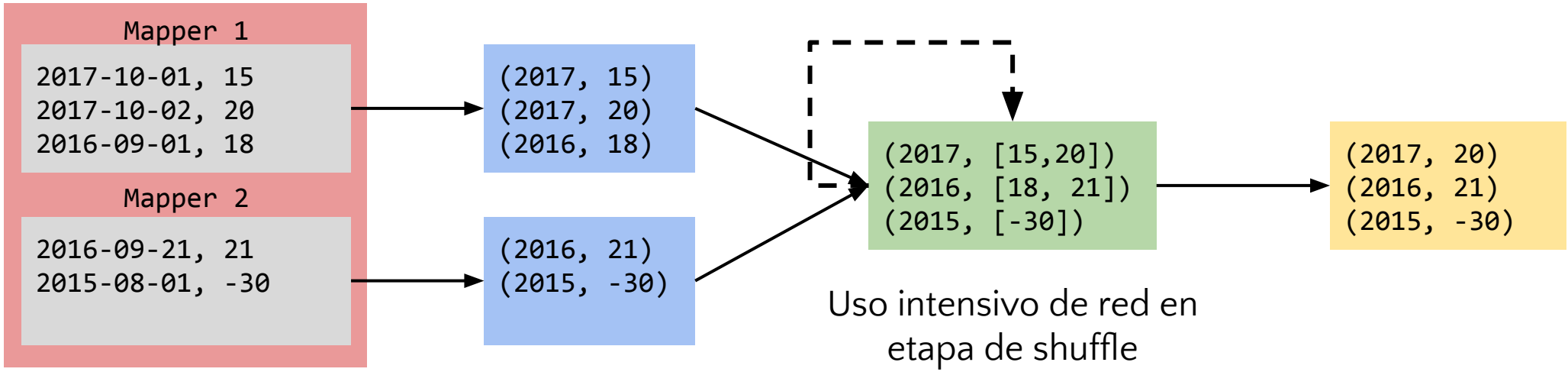
Map Reduce



Map Reduce



Map Reduce



Map Reduce

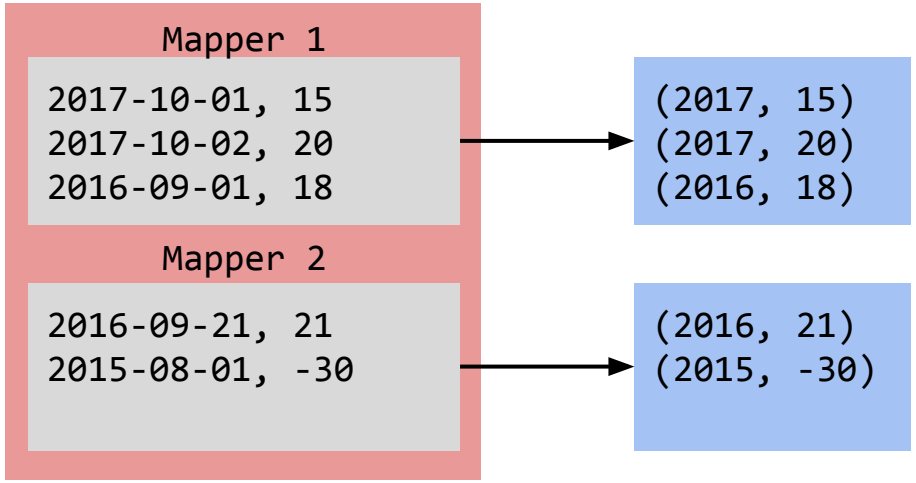
Mapper 1

```
2017-10-01, 15
2017-10-02, 20
2016-09-01, 18
```

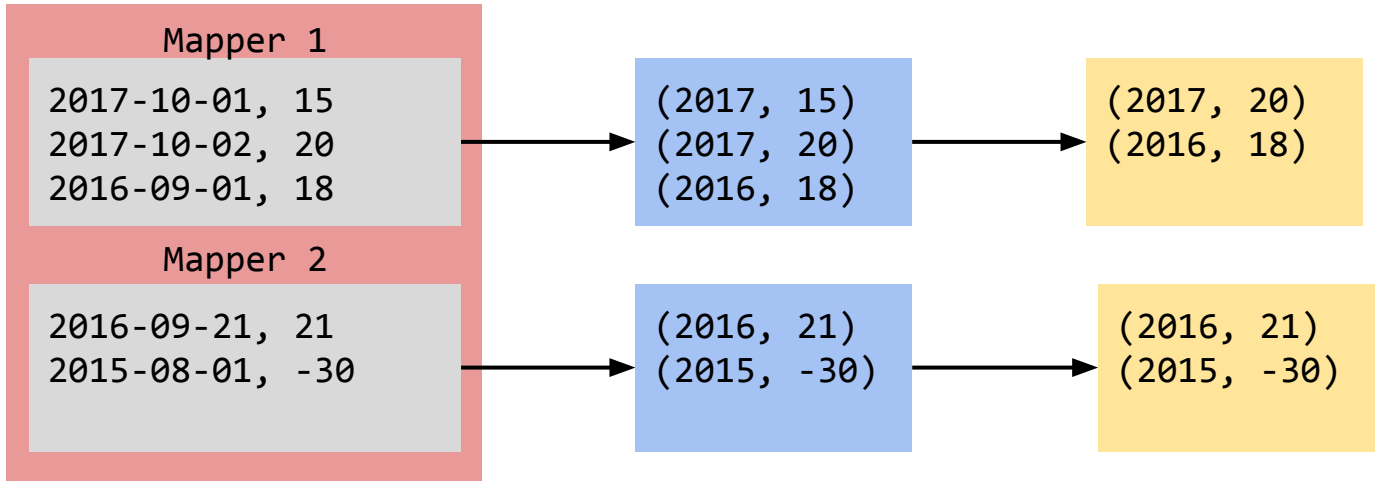
Mapper 2

```
2016-09-21, 21
2015-08-01, -30
```

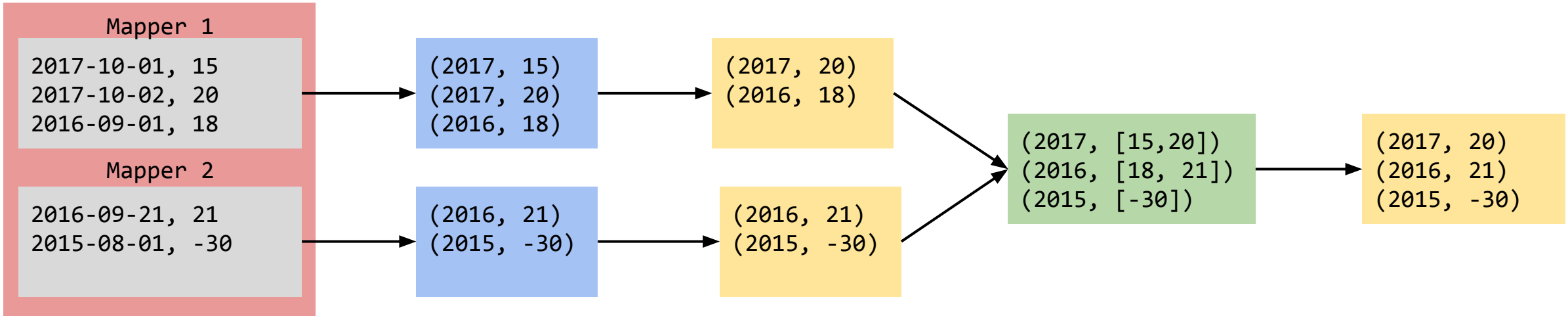
Map Reduce



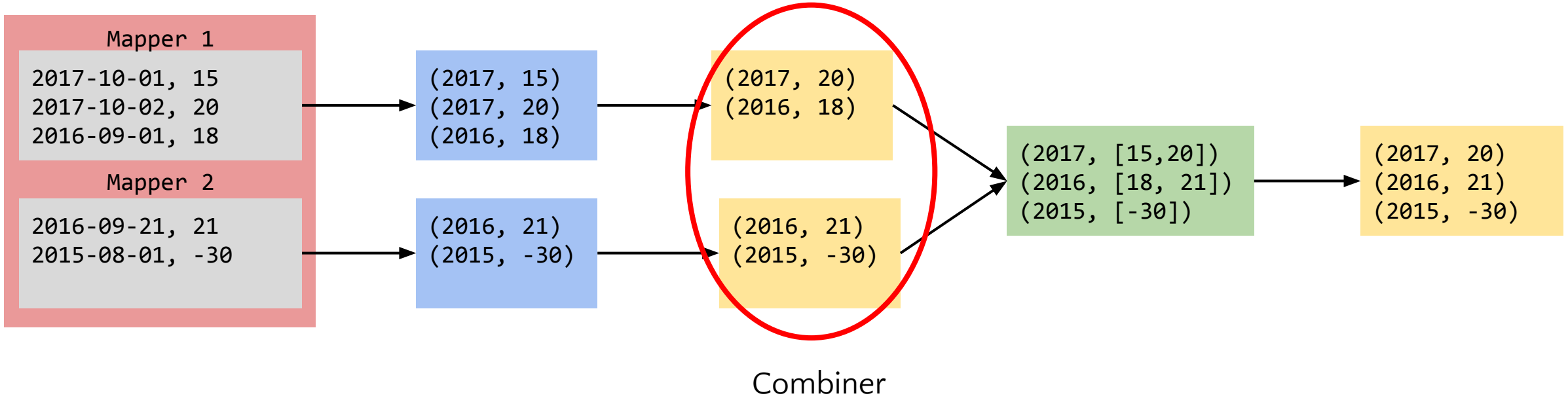
Map Reduce



Map Reduce



Map Reduce

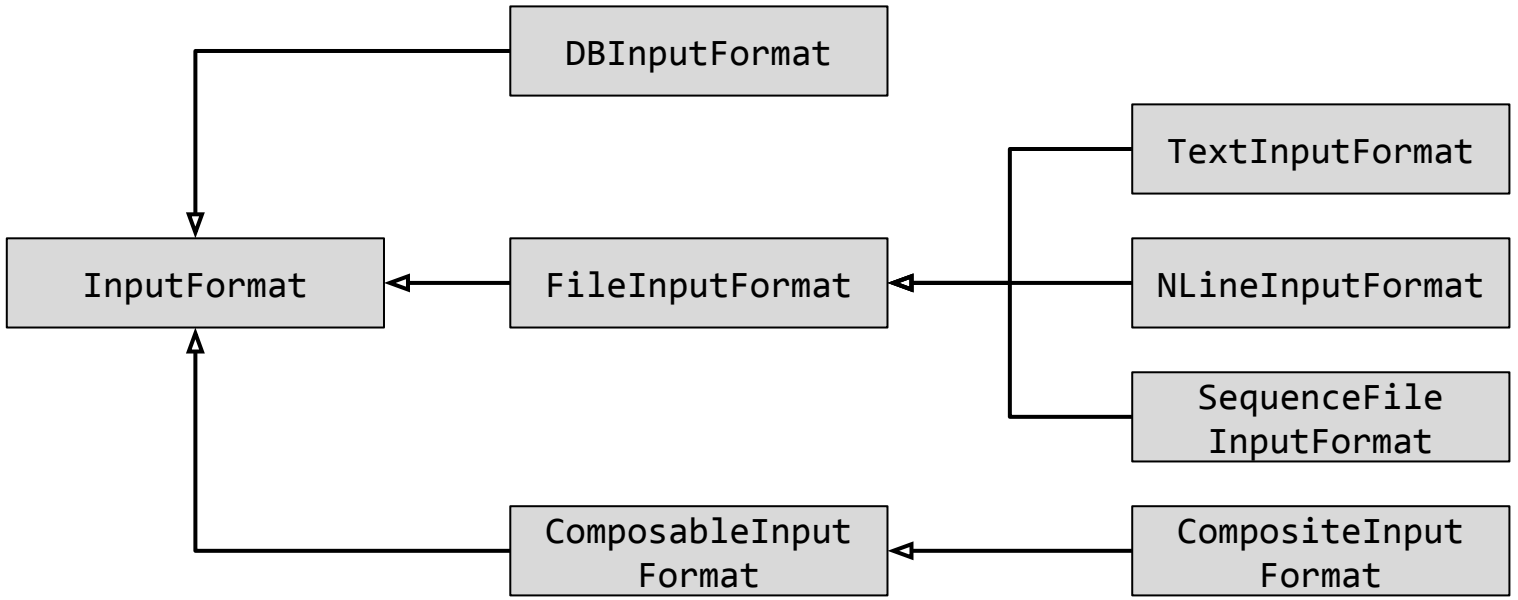


Map Reduce (I/O)

Los jobs map reduce por defecto tienen como entrada: **<LongWritable, Text>**

Esto corresponde al offset en bytes de una linea de texto y la linea que comienza en ese offset.

Existe un árbol de clases de donde se puede extender y configurar diferentes input formats:



Map Reduce (I/O)

Como salida de un mapper, también pueden configurarse clases custom para serializar y deserializar implementando de **`WritableComparable`**

```
public class Pasajero implements WritableComparable<Pasajero> {
     public Text name;
     public IntWritable edad;

     @Override
     public void readFields(DataInput input) throws IOException {
          this.name.readFields(input);
          this.edad.readFields(input);

     }
     @Override
     public void write(DataOutput output) throws IOException {
          this.name.write(output);
          this.edad.write(output);
     }
     @Override
     public int compareTo(Pasajero o) {
          return this.name.compareTo(o.name);
     }
}
```

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES