Conceptos y Aplicaciones de Big Data

MAPREDUCE

MÚLTIPLES JOBS

Prof. Waldo Hasperué
whasperue@lidi.info.unlp.edu.ar

Temario

Problemas que requieren más de un job.

Ejemplo

¿Cómo calculamos el desvío estándar?

$$s^2 = \frac{\sum_{i=1}^{n} (x_i - \overline{x})^2}{n-1}$$

Ejecutando varios jobs

Muchas veces, resolver un problema complejo representa ejecutar varios jobs, uno detrás de otro de manera secuencial:

Job1 ➔ Job2 ➔ Job3 ➔ Job4

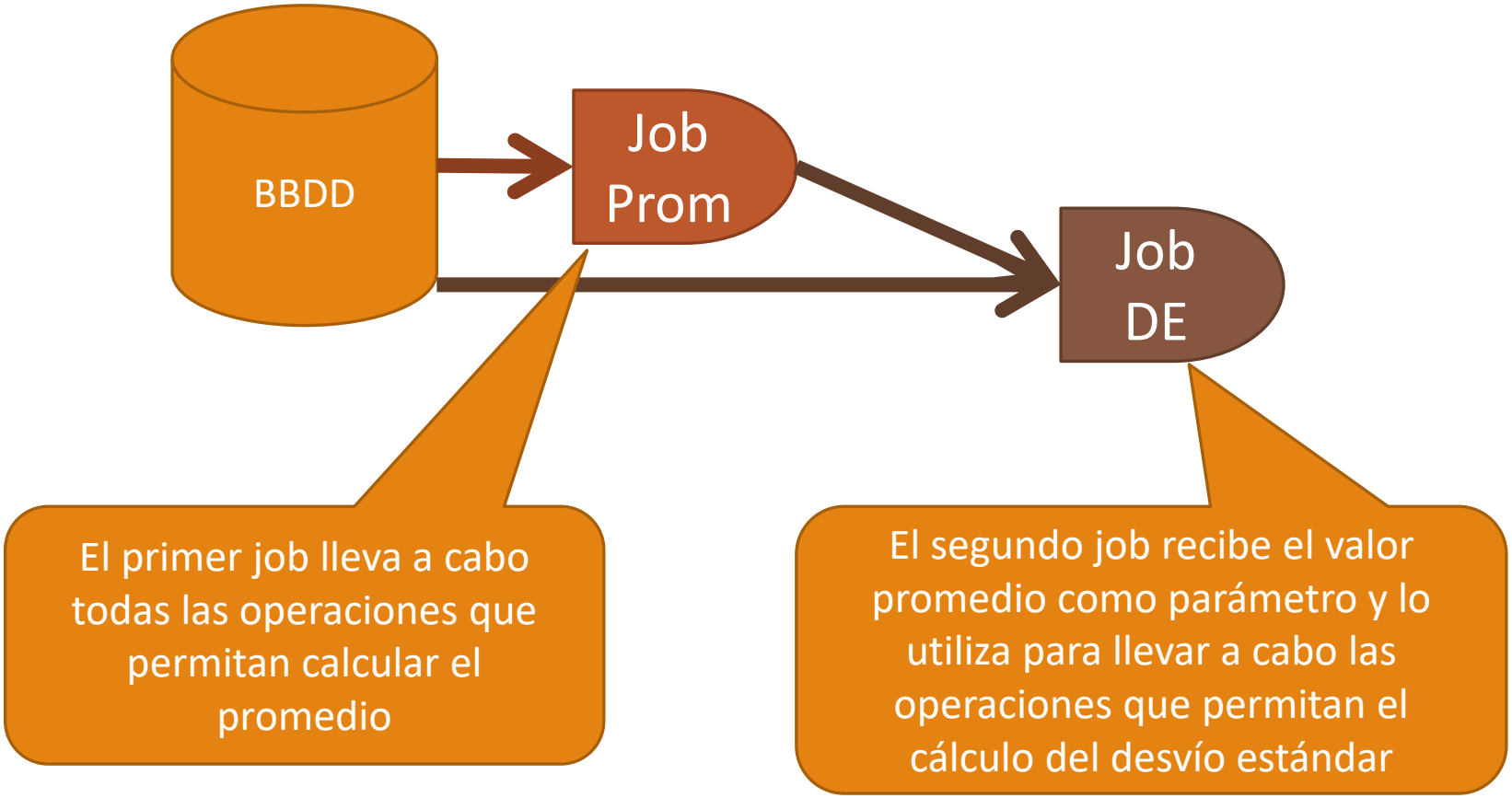
La salida del Job *i* es entrada para el Job *i*+1. La salida del último Job es el resultado final para el usuario.

Ejecutando varios jobs

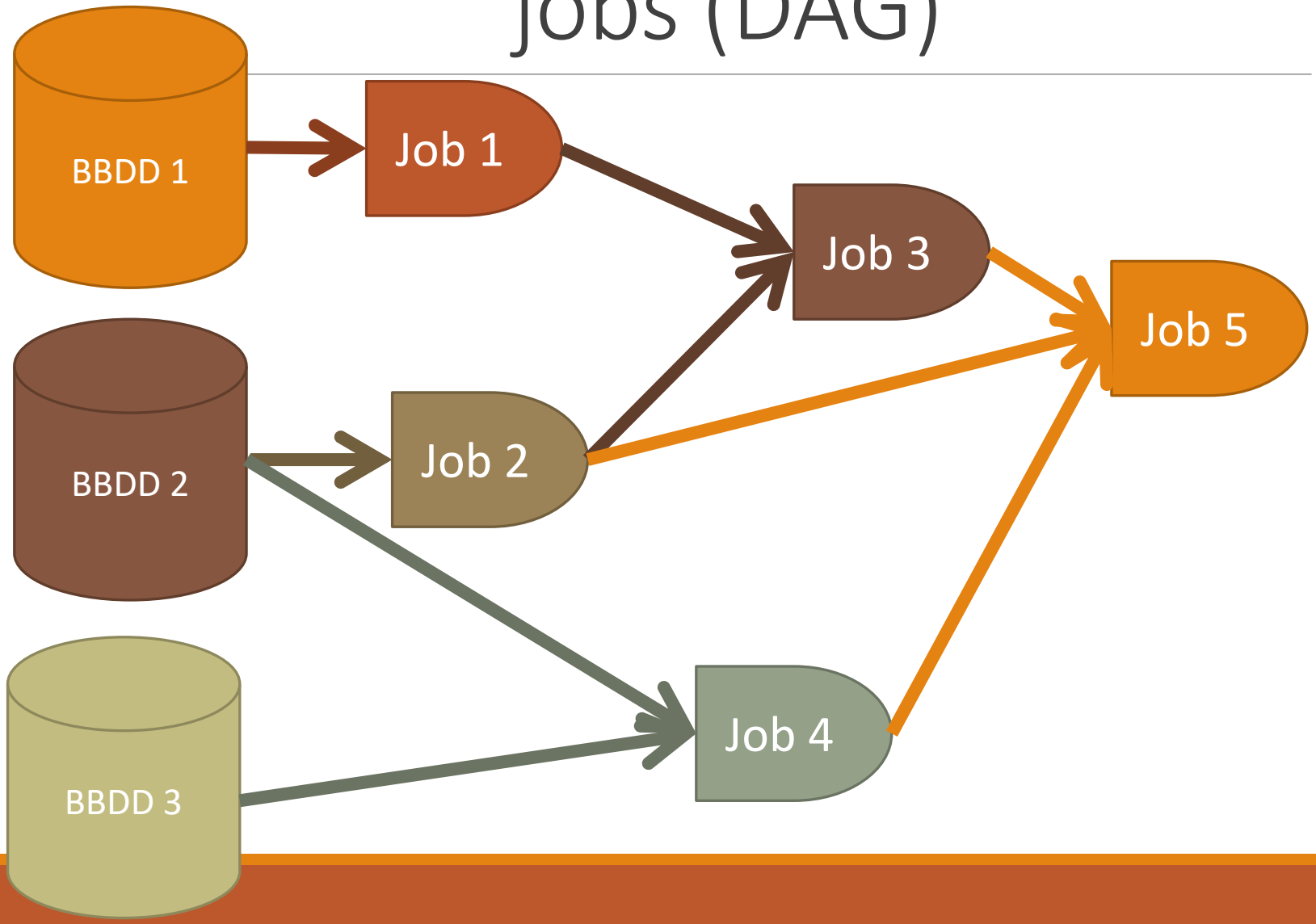
A cada Job se le configura un *mapper* y un *reducer* (y eventualmente un *combiner*).

Opcionalmente, dependiendo del problema, dos o más Jobs podrían ejecutar el mismo *mapper* y/o el mismo *reducer* (la misma implementación).

Ejemplo - Desvío estándar Proceso de varios jobs (DAG)



Ejemplo - Proceso de varios jobs (DAG)



Ejemplo de varios jobs

Se posee un dataset de un sitioweb con el siguiente log de la actividad de sus usuarios:

<id_user, id_page, time>

Se desea realizar un programa MapReduce que devuelva para cada usuario, que página fue la más visitada (la página en la que más tiempo permaneció).

Ejemplo de varios jobs

Id_user	Id_page	time
1	3	10
1	5	45
1	3	23
2	2	12
2	3	20
2	2	13
2	4	15

Ejemplo de varios jobs

Job 1 - Solución 1

```
def fmap (key, value, context):
      id_user = key
      data = value.split("\t")
      id_page = data[0]
      time = data [1]

      context.write(id_user, (id_page, time))
```

El reducer debería llevar un acumulado para cada página, para luego buscar el máximo

Ejemplo de varios jobs

Job 1 - Solución 2

```
def fmap (key, value, context):
      id_user = key
      data = value.split("\t")
      id_page = data[0]
      time = data [1]

      context.write(id_page, (id_user, time))
```

El reducer recibe todos los usuarios de una misma página.
Pero luego no podemos determinar la más visitada para cada usuario.

Ejemplo de varios jobs

Job 1 - Solución 3

```
def fmap (key, value, context):
        id_user = key
        data = value.split("\t")
        id_page = data[0]
        time = data [1]

        context.write((id_user, id_page), time)
```

Para cada usuario y cada página se puede sumar el tiempo total

Ejemplo de varios jobs

Job 1 - Solución 3

```
def fred (key, values, context):
      id_user, id_page = key
      total = 0
      for v in values:
            total+= v

      context.write((id_user, id_page), total)
```

Ejemplo de varios jobs

Job 1 - Salida

Id_user, id_page	time
1 3	33
1 5	45
2 2	25
2 3	20
2 4	15

Ejemplo de varios jobs

Job 2

```
def fmap (key, value, context):
      id_user = key
      data = value.split("\t")
      id_page = data[0]
      time_acum = data[1]

      context.write(id_user,(id_page, time_acum))
```

Para cada usuario se puede calcular la página con mayor tiempo acumulado.

Ejemplo de varios jobs

Job 2

```
def fred (key, values, context):
      maxTime = -1
      for v in values:
            id_page = v[0];    time_acum = v[1]
            if time_acum > maxTime:
                  maxTime = time_acum
                  maxPage = id_page
      context.write(key,(max_page, max_time))
```

Para cada usuario se obtiene la página con mayor tiempo acumulado.

Ejemplo de varios jobs

Job 2 - Salida

id_user	id_page	time_acum
1	5	45
2	2	25

Ejemplo de varios jobs

```
job1 = Job(inputDir, tmpDir, fmap1, fred1)
success = job1.waitForCompletion()
```

```
job2 = Job(tmpDir, outputDir, fmap2, fred2)
success = job2.waitForCompletion()
```