# Conceptos y aplicaciones en Big Data

## Práctica 2 - Hadoop MapReduce

1) ¿En el dataset del ejercicio 1 de la práctica 1 indique para cada Job, si se vería beneficiado por una función combiner? En caso afirmativo, ¿cuál es la implementación de dicha función? ¿Qué datos recibe cada reduce, al utilizar la función combiner?

2) Implemente una función combiner para el problema del WordCount.

3) Implemente un job MapReduce para calcular el máximo, mínimo, promedio y desvío stándard de las ocurrencias de todas las palabras del dataset Libros.

4) Utilice el dataset Libros para implementar una aplicación MapReduce que devuelva como salida todos los párrafos que tienen una longitud mayor al promedio.

5) El dataset website tiene información sobre el tiempo de permanencia de sus usuarios en cada una de las páginas del sitio. El formato de los datos del dataset es:

   <id_user, id_page, time>

   Implemente una aplicación MapReduce, utilizando combiners en los casos que considere necesario, que calcule

   a. La página más visitada (la página en la que más tiempo permaneció) para cada usuario
   b. El usuario que más páginas distintas visitó
   c. La página más visitada (en cuanto a cantidad de visitas, sin importar el tiempo de permanencia) por todos los usuarios.

   Indique como queda el DAG del proceso completo (las tres consultas)

6) Cómo plantearía una solución MapReduce a los siguientes algoritmos secuenciales:

a.

i. entrada

```
textos: array [1..N] of string (dataset libros)
```

ii. algoritmo

```
a={}; b={}; N = len(textos)
for l in textos:
    words = l.split()
    for w in words:
        a[w] = a[w]+1
for w in a.keys():
    for l in lines:
        words = l.split()
        if w in words:
            b[w]=b[w]+1
for k in a.keys():
    print(k + " = " + str(a[w] * (N / b[w])))
```

b.

i. entrada

datos: array [1..N] of <$int_1$, $int_2$, ..., $int_M$>
(todos los valores están dentro de un rango de valores conocido, para poder usarlos como índices del tensor)

ii. algoritmo

```
for t in datos:
     v = t.split("\t")
     c = v[-1]
     for a in range(len(v)-1):
          x= v[a]
          m[a][x][c] = m[a][x][c] + 1

max=[[0,0,0], [0,0,0]]
for x in range(len(m)):
     for y in range(len(m[0])):
          for z in range(2):
               if(m[x][y][z] > max[z][0]):
                    max[z][0] = m[x][y][z]
                    max[z][1]=x
                    max[z][2]=y

for z in range(2):
     print(z +";" + max[z][1] +";" + max[z][2])
```