Programación de Objetos Distribuidos
Trabajo Práctico Especial

Se cuenta con un subconjunto del dataset del **Censo Nacional de Población, Hogares y Viviendas 2010: Censo del Bicentenario**[1]. En un archivo CSV, cada una de las filas representa un habitante registrado en el censo. Cada fila está compuesta por los siguientes campos:

- **`condicionActividad`**: Condición de actividad del habitante, siendo los valores posibles:
  - 0: Sin Datos
  - 1: Ocupada
  - 2: Desocupada
  - 3: Económicamente inactiva
- **`hogarId`:** Identificador del hogar donde habita.
- **`nombreDepartamento`**: Nombre del departamento del hogar donde habita.
- **`nombreProvincia`:** Nombre de la provincia del hogar donde habita.

Se utilizará el concepto de Regiones Integradas de Argentina[2] donde se constituyen cinco regiones del país, como se indica en el siguiente mapa:

[1] Fuente: http://datar.noip.me/dataset/censo-2010-microdatos
[2] https://es.wikipedia.org/wiki/Regiones_integradas_(Argentina)

Requerimientos

Se requiere generar una aplicación de consola que utilice el modelo de programación MapReduce junto con el framework HazelCast y resuelva las siguientes queries:

1. **Total de habitantes por región del país, ordenado descendentemente por el total de habitantes.**

Por ejemplo:

```
Región Buenos Aires,461683
Región del Norte Grande Argentino,206035
Región Centro,192909
Región del Nuevo Cuyo,79086
Región Patagónica,60287
```

2. **Los "n" departamentos más habitados de la provincia "prov".**

Por ejemplo, con **n=10** y **prov=Santa Fe**:

```
Rosario,29689
La Capital,12946
General López,4736
General Obligado,4465
Castellanos,4396
San Lorenzo,3875
Las Colonias,2580
Caseros,2144
Constitución,2092
San Jerónimo,2005
```

3. **Índice de desempleo por cada región del país, ordenado descendentemente por el índice de desempleo, donde el índice se calcula a partir de la siguiente fórmula, mostrándose siempre dos decimales:**

$$I_{Desempleo} = \frac{Total\ Desocupados}{(Total\ Ocupados + Total\ Desocupados)}$$

Por ejemplo:

```
Región del Nuevo Cuyo,0.07
Región del Norte Grande Argentino,0.06
Región Patagónica,0.06
Región Buenos Aires,0.06
Región Centro,0.06
```

4. **Total de hogares por cada región del país ordenado descendentemente por el total de hogares.**

Por ejemplo:

```
Región Buenos Aires,442414
Región del Norte Grande Argentino,195524
Región Centro,184627
Región del Nuevo Cuyo,75449
Región Patagónica,57713
```

5. **Por cada región del país el promedio de habitantes por hogar, ordenado descendentemente por el promedio de habitantes, mostrándose siempre dos decimales.**

Por ejemplo:

```
Región del Norte Grande Argentino,1.05
Región del Nuevo Cuyo,1.05
Región Centro,1.04
Región Patagónica,1.04
Región Buenos Aires,1.04
```

6. **Los nombres de departamentos que aparecen en al menos "n" provincias, ordenado descendentemente por el número de apariciones**

Por ejemplo, con **n=5**:

```
Capital,11
San Martín,5
Rivadavia,5
25 de Mayo,5
```

7. **Los pares de provincias que comparten al menos "n" nombres de departamentos, ordenado descendentemente por la cantidad de coincidencias. Aclaración: en la salida se debe mostrar el par de nombres de provincias ordenados alfabéticamente.**

Por ejemplo, con **n=4**:

```
Buenos Aires + Chaco,7
Buenos Aires + Corrientes,5
San Juan + Santiago del Estero,4
Corrientes + Mendoza,4
Buenos Aires + Río negro,4
Buenos Aires + Mendoza,4
Buenos Aires + San Luis,4
```

Consideraciones

Cada corrida de la aplicación realiza una de estas queries sobre los datos obtenidos a partir de un archivo CSV que se provee con la estructura indicada al comienzo (no es necesario validarla).

Para medir performance, se deberán escribir en un archivo los *timestamp* de los siguientes momentos:

- Inicio de la lectura del archivo de entrada
- Fin de lectura del archivo de entrada
- Inicio de un trabajo MapReduce
- Fin de un trabajo MapReduce (incluye la escritura del archivo de respuesta)

Todos estos momentos deben ser escritos en la salida luego de la respuesta con el timestamp en formato: `dd/mm/yyyy hh:mm:ss:xxxx` y deben ser claramente identificables.

Ejemplo del archivo de tiempos:

```
23/10/2017 14:43:09:0223 INFO  [main] Client (Client.java:76) - Inicio de
la lectura del archivo
23/10/2017 14:43:23:0011 INFO  [main] Client (Client.java:173) - Fin de
lectura del archivo
23/10/2017 14:43:23:0013 INFO  [main] Client (Client.java:87) - Inicio del
trabajo map/reduce
09/11/2017 14:43:23:0490 INFO  [main] Client (Client.java:166) - Fin del
trabajo map/reduce
```

La información de cuál es la query a correr, cuáles son los archivos involucrados y los parámetros necesarios se reciben a través de argumentos de línea de comando al llamar a la aplicación.

Por ejemplo:

```
$> java -Daddresses=xx.xx.xx.xx;yy.yy.yy.yy -Dquery=1 -DinPath=censo.csv
-DoutPath=output.txt -DtimeOutPath=time.txt [queryParams] client.MyClient
```

donde

- `xx.xx.xx.xx;yy.yy.yy.yy` son las direcciones IP de los nodos,
- censo.csv es el archivo de entrada con los datos a procesar
- output.txt es el archivo de salida con los resultados de la query
- time.txt es el archivo de salida con los timestamp de los tiempos de la lectura del archivo de entrada y de los trabajos map/reduce
- `[queryParams]`:
  - `-Dn=XX` para la queries 2, 6 y 7
  - `-Dprov=XX` para la query 2
  - Vacío para las otras queries

De esta forma,

```
$> java -Daddresses=10.6.0.1;10.6.0.2 -Dquery=6 -DinPath=censo.csv
-DoutPath=output.txt -DtimeOutPath=time.txt -Dn=5 client.MyClient
```

escribe en el archivo **output.txt** los nombres de los departamentos que aparecen en al menos 5 provincias, según los datos presentes en `censo.csv`, utilizando los nodos `10.6.0.1` y `10.6.0.2` para su procesamiento. Escribe además en el archivo **time.txt** los *timestamp* de inicio y fin de la lectura del archivo y de los trabajos map/reduce.

El nombre del cluster y los nombres de los mapas de Hazelcast a utilizar en la implementación deben contener los números de legajo de los integrantes del grupo para así evitar conflictos con los mapas y poder hacer pruebas de distintos grupos en simultáneo. Se debe utilizar la siguiente convención:

```
legajo1-legajo2-legajo3-legajo4
```

La aplicación debe entonces correr la query y escribir en un archivo la respuesta a la query.

La implementación debe respetar exactamente el formato de salida enunciado.

Condiciones del trabajo práctico

- El trabajo práctico debe realizarse en grupos de a cuatro personas.
- Cada una de las opciones debe ser implementada como uno o más job MapReduce que pueda correr en un ambiente distribuido utilizando un grid de Hazelcast.
- Los componentes del job, clases del modelo, test y el diseño de cada elemento del proyecto queda a criterio del equipo, pero debe estar enfocado en:
    - Que funcione correctamente en un ambiente concurrente, MapReduce en hazelcast.
    - Que sea eficiente para un gran volumen de datos, particularmente en tráfico de red
    - Mantener buenas prácticas de código como comentarios, reutilización, legibilidad y mantenibilidad.
- En **campus.itba.edu.ar,** en la sección Contenido / Evaluación / TPE, se irán dejando subconjuntos del dataset del Censo con diferentes cantidades de habitantes para así poder pobar con diversos volúmenes de datos.

Se debe entregar

- El ***código fuente*** de la aplicación:
    - Utilizando el arquetipo de Maven utilizado en las clases.
    - Con una correcta separación de las clases de cliente y servidor.
- Un ***documento*** explicando:
    - Como preparar el entorno a partir del código fuente para ejecutar la aplicación en un ambiente con varios nodos.

- Brevemente como se diseñaron los componentes de cada trabajo MapReduce, qué decisiones se tomaron y con qué objetivos. Además alguna alternativa de diseño que se evaluó y descartó, comentando el porqué.
- El análisis de los tiempos cada proceso corriendo en clusters variando la cantidad de nodos e indicando cuál sería la cantidad de nodos mejor para cada uno (analizando clusteres de hasta 6 nodos), sobre un dataset a determinar por la cátedra.
- No se deben entregar los binarios para ejecutar.

Entregas

- **Hasta el día 30/10/2017** se deben inscribir los nombres de los integrantes de cada equipo en la sección **Grupos** de Campus ITBA
- **Hasta el 07/11/2017 a las 12 hs** deben cargar el ***código fuente*** y el ***documento*** del trabajo práctico especial en la actividad "Entrega TPE" localizada en la sección Contenido / Evaluación / TPE de Campus ITBA.
- **El día 13/11/2017 a las 18hs** cada grupo tendrá 15 minutos para configurar su entorno y mostrar la ejecución de la aplicación a la cátedra realizando dos corridas sobre una misma configuración de nodos a utilizar, con archivos (datasets) diferentes a determinar por la cátedra. Se tomará nota de las respuestas y los tiempos de ejecución y esto será parte de la evaluación del trabajo. A criterio de la cátedra también se podrán realizar preguntas sobre la implementación de los mismo.
- **El día del recuperatorio seŕa el 27/11/2017.** Para aquellos equipos que lo requieran, se les podrá pedir que corrijan o agreguen algún elemento o query al proyecto.
- ***No se aceptarán entregas fuera de los días y horarios establecidos.***