Arquitectura i tecnologies del software
Pràctica (I) - BigData

Enginyeria Informàtica
Escola d'Enginyeria (EE)
Universitat Autònoma de Barcelona (UAB)

Lluís Gesa

Dept. Ciències de la Computació (DCC)

v1.0

2018

Contents

1 Introducció

El present document descriu la part 1 de la pràctica proposada per posar en pràctica certs coneixements adquirits durant el temari **BigData** en relació a ***Map-Reduce*** .

2 Entorn de treball

No hi ha gaires restriccions en relació al llenguatge per implementar la pràctica. Pot ser: **Java**^{tm} , **Python**, **C**, **C++**, **Perl**, **Ruby**. Simplement us heu d'assegurar que és compilable i executable en un shell de Linux/GNU (que serà l'entorn d'avaluació), és a dir, si es realita el treball en un entorn de desenvolupament concret us heu d'assegurar que podeu generar una exportació del projecte per ser compilable des de línia de comandes de forma fàcil.

Hi ha la opció d'entregar el projecte en format **Eclipse** .

No podeu emprar cap llibreria externa ni entorn preinstal.lat al sistema més enllà de les eines del propi llenguatge.

3 Enunciat

Map-Reduce ha sigut la metodologia de procés de dades que va iniciar la revolució del concepte **BigData** , tot i que actualment ja hi ha altres opcions molt més eficients. El seu ús continua essent plenament vigent, viu i molt estès.

Se us proposa que com enginyers de software que sou, penseu en una petita aplicació que implementi el proces ***Map-Reduce*** des de 0 i s'apliqui a una customització per fer l'histograma de paraules dels arxius de text pla passats per paràmetre. El concepte fer-ho des de 0 vol dir: no podeu usar cap llibreria o plataforma existent que implementi re relacionat amb ***Map-Reduce*** . Per exemple en **Java**^{tm} podeu usar els contenidors Vectors, Hash, que us proporciona el llenguatge però cap funcionalitat més que apliqui funcions sobre cada element contingut ni re semblant.

Per obtenir la màxima versatilitat de la implementació ***Map-Reduce*** , hauríeu de crear threads i paral.lelitzar el procés.

Se us facilita un exemple d'arxiu *ArcTecSw_2018_BigData_Practica_Part1_Sample.txt* i el corresponent resultat *ArcTecSw_2018_BigData_Practica_Part1_Sample_Result.txt*

4 Objectius

Un apunt inicial important, no es considerarà cap pràctica com susceptible d'aprovar si no compila ni executa.

4.1 Obligatoris

1. El primer objectiu és fer una implementació de l'algoritme ***Map-Reduce*** . No si valen aproximación i fer presumpcions. L'algoritme (que el podeu trobar en els apunts i articles de suport que se us facilita), és molt clar en les 2 fases que disposa: El Map i el Reduce, així com quina sortida ha de generar cada fase.

2. Adaptar les fases de Map i Reduce per comptar les paraules d'un arxiu donat o varis arxius, i mostrar el resultat per pantalla.

3. Implementar el codi en el llenguatge escollit. L'aplicació ha de poder ser cridada amb 1 o més arxius de text : TextCounter File1.txt File2.txt. El resultat s'ha de volcar a pantalla separats per arxiu:

```
File1.txt:
a : 1
hola : 1
punt : 2
  ...

File2.txt:
the : 1
fire : 2
room : 1
  ...
```

   El format de la crida i el resultat s'ha de respectar ja que s'executarà el sistema amb scripts automàtics per detectar-ne la validesa. Si el vostre aplicatiu rep més parametres que permeten configurar coses, genereu un script "TextCounter" que cridi com toca el vostre aplicatiu amb parametres per defecte per exemple.

   Tingueu en compte temes de reserva de memòria, que l'aplicació pugui treballar amb un arxiu de Gbs sense que salti per l'aire per out-of-memory.

4. Com actueu com enginyers de software, caldrà entregar un document (o documents): d'anàlisi i disseny, explicant la vostra proposta d'implementació i treball realitzat. Haureu de fer ús de UML per explicar classes, seqüències. Un manual/secció d'usuari, com és compila. I finalment una explicació de test exposant les proves que heu realitzat, e indicant llistats/gràfiques de performance comparant varies execucions modificant-hi paràmetres de configuració (més o menys processos paral.lels, spplit de les dades), comparacions amb un loop lineal clàssic de comptar paraules, etc.. etc.

5. El codi i l'estructura del projecte ha de ser net, clar i pensat. El codi ha de disposar de comentaris aclarint la seva funcionalitat e indicant clarament els components del grup autor.

4.2 Opcionals

Els objectius opcionals proposats serviran per pujar nota en cas de dubte, sempre hi quan els punts obligatoris estigui completat.

Bàsicament com a opcional es contempla afegir funcionalitat:

- Opció de fer histograma de lletres enlloc de paraules.
- Opció de fer un histograma comú amb tots els arxius passats enlloc d'un histograma per arxiu.

5 Entrega

Per fer l'entrega caldrà que m'envieu un email amb el zip de tot el projecte amb el nom del zip : **ats_2018_BD_NUMERO_DE_GRUP_pract1.zip**.

Dins el zip hi ha d'haver:

- doc : Directori amb la documentació d'arxiu(s)
- bin/TextCounter : Executable que rebrà per paràmetres el nom(s) d'arxiu(s)
- src : Directori amb tot el codi per a compilar

El numero de grup se us assignarà quan es formin els equips.

Feu-vos una còpia per vosaltres per si hi ha algun problema. Dins el ZIP hi ha d'haver un executable ja compilat (si cal compilació), i l'estructura de projecte necessari per a compilar-lo. Així com la documentació demanada. No afegiu arxius de test d'entrada/sortida (és a dir, no entregueu un zip de varis centenars de Mb!!)

6 Avaluació

Elements que es tindràn en compte per avaluar:

- Es respecten els formats i noms de les coses: Nom de zip, contingut, format de les crides, del resultat...
- La documentació està treballada, no simplement 2 pàgines.
- El codi es net, pulit i entendor
- NO HI HA CAP ELEMENT COPIAT NI PLAGIAT, ni de companys ni d'internet: La còpia o plagi suposa suspendre TOT ATS.
- Hi ha gestió d'errors.
- Per suposat es compleix el que es proposa fer.