Sistemes Operatius II - Pràctica 2

Setembre del 2019

Índex

1 Introducció

A l'assignatura de Sistemes Operatius II es realitzarà un únic projecte pràctic al llarg del curs. L'objectiu general del projecte pràctic és desenvolupar una aplicació (sense interfície gràfica) que permeti extreure i indexar les paraules contingudes en múltiples fitxers de text pla.

Aquests fitxers de text són llibres electrònics gratuïts en anglès i provenen del web Gutenberg (http://www.gutenberg.org/). L'aplicació haurà de llegir cadascun d'aquests fitxers de text, extreure les paraules que contenen aquests, i indexar les paraules en un arbre fent servir un diccionari de paraules (pràctica 2). Un cop s'ha obtingut aquesta informació s'haurà de poder desar i llegir la l'arbre a disc (pràctica 3). Finalment, es processaran múltiples fitxers en paral·lel fent servir múltiples processos (pràctica 4) i múltiples fils (pràctica 5).

Les pràctiques es realitzen en parella i podeu discutir amb els vostres companys la solució que implementeu. En tot cas, s'espera que cada parella implementi el seu propi codi.

Les pràctiques estan encavalcades entre sí. És a dir, per a realitzar una pràctica és necessari que la pràctica anterior funcioni correctament. Assegureu-vos doncs que les pràctiques estan dissenyades de forma que puguin ser ampliades de forma fàcil. Per tal d'aconseguir-ho es donarà, per a cada pràctica, unes pautes a seguir per aconseguir-ho. A més, en finalitzar la pràctica 3 es donarà la solució perquè es puguin implementar la pràctica 4 i 5. L'estudiant serà lliure de seguir amb la seva implementació o bé amb la solució proporcionada pels professors.

2 La pràctica

L'objectiu d'aquesta pràctica se centra en la manipulació de punters i la lectura de fitxers de text. Caldrà manipular estructures de dades i per minimitzar la feina relacionada es proporciona el codi necessari. Específicament, les tasques a realitzar en aquesta pràctica són:

1. Es proporciona un fitxer de text pla que és un diccionari de paraules en anglès. A cada línia del fitxer hi ha una paraula diferent. Caldrà llegir aquest fitxer i inserir les paraules a una estructura d'arbre. Els detalls d'aquest punt s'expliquen a la secció 2.1.

2. Es proporciona un fitxer de text pla en què cada línia és el nom d'un fitxer. Cadascun d'aquests fitxers és un fitxer de text pla de la base de dades Gutenberg. L'aplicació llegirà cadascun d'aquests fitxers, n'extraurà les paraules que contenen, i inserirà la informació associades a paraules de diccionari a l'arbre descrit abans. Veure secció 2.2 per a més informació.

Els dos punts anteriors es descriuen a continuació. A la secció 3 es mencionen unes pautes a seguir per implementar aquesta pràctica. D'aquesta forma podeu estructurar el programa en funció de la feina que haureu de fer més endavant.

2.1 Indexació de les paraules del diccionari en un arbre

Lectura de les paraules de diccionari

A les aules hauríeu de tenir instal·lada una aplicació de terminal anomenada `look` que permet cercar paraules dintre d'un fitxer de diccionari de paraules en anglès. Per exemple, executeu

```
$ look see
```

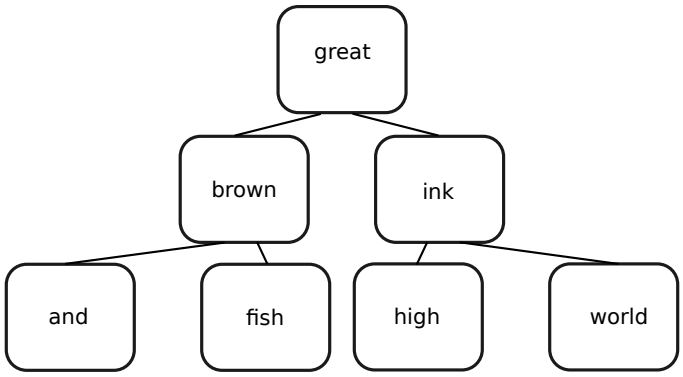

Figura 1: Arbre binari que emmagatzema paraules.

Aquesta instrucció cerca, dintre del fitxer de diccionari, totes les paraules que comencen per `see` i les mostra per pantalla.

Disposeu, juntament amb l'enunciat d'aquesta pràctica, del fitxer de diccionari `words` que utilitza l'aplicació `look`. Aquest fitxer es troba en el directori `src/words`. Les paraules estan separades entre sí per línies (editeu-lo amb un editor de text pla!) i pot contenir paraules amb caràcters en majúscula o minúscula. El fitxer de què disposeu amb aquesta pràctica ha estat netejat per eliminar totes les paraules que contenen accents o dièresis per facilitar el tractament de fitxers de text en llenguatges com el C[1].

Es proporciona el codi `src/llegir-diccionari.c` que llegeix i imprimeix les paraules del diccionari per pantalla. La lectura de dades d'un fitxer és un tema delicat ja que hi ha múltiples funcions que permeten fer-ho. Es disposa, per exemple, de les funcions *fgets* o *fscanf* per fer llegir-ho. S'ha escollir fer servir *fgets*. La raó és que *fscanf* és una funció insegura i pot fer petar el programa. Podeu esbrinar per què?

Indexació de les paraules de diccionari

Les paraules de diccionari s'indexaran en un arbre balancejat. Es proporciona el codi d'un arbre balancejat a `src/arbre-binari`. Aquest arbre està programat per fer servir un sencer com a clau d'indexació. Caldrà adaptar el codi perquè l'arbre faci servir les paraules del diccionari com a clau d'indexació. Per a cada node s'emmagatzemen a l'esquerra les paraules que lexicogràficament són anteriors a la paraula del node, mentre que a la dreta es troben les paraules lexicogràficament posteriors a la paraula del node. A l'hora d'indexar les paraules, no s'ha de diferenciar entre les minúscules i majúscules[2]. Cada node ha d'emmagatzemar la paraula amb la mida mínima necessària (i.e. no es pot suposar que la paraula té una mida màxima de, per exemple, 100 caràcters).

2.2 Extracció i indexació de la informació dels fitxers de text Gutenberg

A continuació es descriu el processament de la base de dades Gutenberg. Com s'ha comentat abans, l'aplicació llegirà cadascun d'aquests fitxers, n'extraurà les paraules que contenen, i inserirà

[1] Típicament les paraules amb accents o dièresis s'emmagatzemen a un fitxer de text fent dos bytes en comptes d'un byte com és el cas de les lletres 'a' a 'z' i 'A' a 'Z'.

[2] No es permetrà la "manipulació" de les cadenes i emmagatzemar a l'arbre les paraules amb tots els caràcters a minúscula o majúscula

```
595
./etext00/00ws110.txt
./etext00/1cahe10.txt
./etext00/1vkip11.txt
./etext00/2cahe10.txt
./etext00/2yb4m10.txt
./etext00/8rbaa10.txt
./etext00/8year10.txt
./etext00/andsj10.txt
./etext00/beheb10.txt
./etext00/benhr10.txt
./etext00/bgita10.txt
./etext00/btowe10.txt
./etext00/cbtls12.txt
./etext00/chldh10.txt
./etext00/cptcr11a.txt
./etext00/cstwy11.txt
./etext00/cyrus10.txt
./etext00/dmsnd11.txt
./etext00/dscmn10.txt
```

Figura 2: Exemple de l'estructura del fitxer `llista.cfg`.

la informació associada a les paraules de diccionari a l'arbre descrit abans.

Base de dades de fitxers de text pla

Al directori `src/base_dades` es proporcionen el conjunt de fitxers de text pla de Gutenberg. El fitxer `llista.cfg` conté un llistat dels fitxers que hi ha. A la figura 2 mostra un exemple de l'estructura d'aquest fitxer: a la primera línia s'emmagatzema el nombre de fitxers a processar. En aquest exemple el valor és 595, tot i que caldrà programar-ho perquè pugui tenir qualsevol valor. A cada línia del fitxer de configuració hi haurà un nom de fitxer. Cadascun d'aquests fitxers és un fitxer de text pla dels quals caldrà extreure les paraules.

Extracció de paraules d'un fitxer

Observeu a la figura 3 un exemple d'un fitxer de text pla a processar. Com es pot veure, el fitxer conté paraules incloent els corresponents signes de puntuació (punt, coma, punt i coma, dos punts, guionet, cometes, signe d'admiració, signe d'exclamació, parèntesi, claudàtors, etc).

Es volen extreure totes les paraules contingudes en els fitxers de text excloent els signes d'admiració així com espais o tabuladors. Així de la primera línia del text de la figura 3 "To sing a song that old was sung," l'aplicació ha d'extreure les paraules "To", "sing", "a", "song", "that", "old", "was", "sung". Cal tenir en compte les següents regles per extreure les paraules:

1. Les paraules poden estar separades per espais, tabuladors o altres signes de puntuació. Es considerarà que aquests símbols no formen part de la paraula. És a dir, en trobar la cadena "why?" s'extraurà la paraula vàlida "why" ja que "?" és un signe puntuació. Les paraules unides per guions, com per exemple "taper-light", són paraules vàlides separades: "taper" i "light". Les paraules que continguin apòstrofs, com per exemple "wit's", són una única paraula vàlida.

2. L'aplicació no té perquè ser capaç de tractar paraules amb accents. Així, paraules com "Wäts" s'ignoraran. Es recomana no intentar tractar aquests casos ja que les lletres amb aquests tipus

```
To sing a song that old was sung,
From ashes ancient Gower is come;
Assuming man's infirmities,
To glad your ear, and please your eyes.
It hath been sung at festivals,
On ember-eves and holy-ales;
And lords and ladies in their lives
Have read it for restoratives:
The purchase is to make men glorious;
Et bonum quo antiquius, eo melius.
If you, born in these latter times,
When wit's more ripe, accept my rhymes,
And that to hear an old man sing
May to your wishes pleasure bring,
I life would wish, and that I might
Waste it for you, like taper-light.
This Antioch, then, Antiochus the Great
Built up, this city, for his chiefest seat;
The fairest in all Syria,
I tell you what mine authors say:
```

Figura 3: Exemple de fitxer de text pla a processar.

de símbols s'emmagatzemen de forma molt particular en un fitxer de text pla i són complicats de processar. Vegeu secció 3 per a més informació al respecte.

3. Paraules que continguin números o altres símbols s'ignoraran. Així, per exemple, una paraula com "hello123" o "##continue" s'ignoraran.

Es proporciona el codi que extreu les paraules d'un fitxer de text, veure codi el directori `src/extraccio_paraules`. Aquest codi, però, no segueix totes les regles esmentades anteriorment. Podeu trobar on és troba el problema?

Indexació de les paraules a l'arbre

L'objectiu és comptar quantes vegades apareix cadascuna de les paraules de diccionari de l'arbre als fitxers de text de Gutenberg.

Donada una paraula extreta d'un fitxer de text (les lletres de les quals han sigut passades a minúscula), caldrà buscar-la a l'arbre. Si hi apareix, s'incrementarà un comptador associada a la paraula. Si la paraula no hi apareix, s'ignorarà.

3 Implementació i planificació

Per tal d'assolir amb èxit aquesta pràctica es recomana revisar abans de tot la Fitxa 1 (Depuració amb gdb i valgrind) i la Fitxa 2 (Recordatori bàsic del llenguatge C i manipulació de cadenes de caràcters). A l'hora de programar és important seguir una estructura modular atès que la resta de pràctiques d'aquesta assignatura es basaran en aquesta primera pràctica.

Es proposa a continuació una forma de procedir per a la implementació d'aquesta pràctica. Tot el codi pot estar situat en el mateix directori.

1. Proveu els codis que se us proporcionen amb aquesta pràctica. En particular, se us proporciona un codi per a) llegir les paraules de diccionari, b) crear un arbre binari balancejat fent servir

com a clau un sencer, b) extreure les paraules d'un fitxer de text. Proveu-los abans de continuar!

2. Modifiqueu el codi perquè es llegeixin les paraules de diccionari i aquestes s'insereixin a l'arbre. Per fer això caldrà modificar (una mica) el codi de l'arbre perquè la clau d'indexació de l'arbre pugui ser una cadena. Us serà particularment útil la funció *strcasecmp*, que permet comparar dues cadenes de caràcters ignorant les majúscules i minúscules. Llegiu atentament al manual d'aquesta funció (la trobareu a Internet, per exemple) per saber com funciona. Cada node haurà d'emmagatzemar la paraula amb la mida mínima necessària (i.e. no es pot suposar que la paraula té una mida màxima).

3. Assegureu-vos que el codi funciona correctament amb el `valgrind`. En particular, que no es facin accessos invàlids a memòria i que tota la memòria s'alliberi correctament.

4. Es proposa que a continuació només s'indexin les paraules d'un únic fitxer de text, no pas tota la base de dades. Baseu-vos en el codi del directori `src/extraccio_paraules` per implementar la funcionalitat indicada a la Secció 2.2. Tingueu en compte que el codi proporcionat no implementa correctament totes les regles d'extracció esmenades en aquesta secció. Arregleu doncs el codi perquè funcioni bé.

5. De nou, assegureu-vos que el codi funciona correctament amb el `valgrind`.

6. Finalment, modifiqueu el codi perquè el paràmetre d'entrada sigui fitxer amb el llistat de fitxers a processar, `llistat.cfg`. Per fer proves podeu fer servir llistats més petits com, per exemple, `llistat_10.cfg` o `llistat_2.cfg`. Aquest fitxer conté els noms de fitxers dels quals caldrà extreure les paraules i indexar-les a l'arbre. Per simplificació podeu suposar que els fitxers a processar estan situats en el mateix directori de l'executable.

7. Assegureu-vos que el codi funciona correctament amb el `valgrind`.

4 Entrega

El fitxer que entregueu s'ha d'anomenar `P2_NomCognom1NomCognom2.tar.gz` (o `.zip`, o `.rar`, etc), on `NomCognom1` és el nom i cognom del primer component de la parella i `NomCognom2` és el cognom del segon component de la parella de pràctiques. El fitxer pot estar comprimit amb qualsevol dels formats usuals (`tar.gz`, `zip`, `rar`, etc). Dintre d'aquest fitxer hi haurà d'haver dues carpetes: `src`, que contindrà el codi font, i `doc`, que contindrà la documentació addicional en PDF. Aquí hi ha els detalls per cada directori:

- El directori `doc` ha de contenir un document (tres o quatre o pàgines, en format PDF, sense incloure la portada) explicant:
  - A l'enunciat de la pràctica es comenta que la funció *fgets* és més segura que *fscanf* per llegir text pla. Podeu comentar per què? Quines avantatges aporta la funció *fgets* a nivell de seguretat?
  - Com s'ha adaptat el codi de la manipulació de l'arbre perquè la clau d'indexació sigui una cadena. Indiqueu de forma clara quines funcions s'han adaptat. Recordeu que, tal com s'indica a la secció 2.1, cada node ha d'emmagatzemar la paraula amb la mida mínima

necessària (i.e. no es pot suposar que la paraula té una mida màxima de, per exemple, 100 caràcters). Podeu incloeu codi per facilitar la descripció de les modificacions realitzades.

- Quin és el problema que té el codi d'extracció de paraules de la secció 2.2? Comenteu breument com ho heu arreglat.
- En cas que incloeu alguna cosa excepcional (diferent) de la demanada a l'enunciat, podeu incloure també l'explicació de la funcionalitat implementada.
- Quines són les 10 paraules que apareixen més cops als fitxers de texts proporcionats? Comenteu quantes vegades apareixen cadascuna d'aquestes 10 primeres paraules i indiqueu clarament com heu obtingut el resultat. Sou capaços d'obtenir el resultat ajudant-vos de les instruccions de la línia de comandes? És a dir, podeu obtenir el resultat sense haver de programar-ho tot en C?

- La carpeta `src` contindrà el codi font comentat (com a mínim les funcions). S'hi han d'incloure tots els fitxers necessaris per compilar i generar l'executable. El codi ha de compilar sota Linux amb la instrucció `make`. Editeu el fitxer `Makefile` en cas que necessiteu afegir fitxers C que s'hagin de compilar.

  Per simplificar, es pot suposar que el codi s'executa dins del directori en què està situada la base de dades. El fitxer de diccionari `words` també estarà situat en aquest directori. El codi només tindrà un únic paràmetre, el nom de fitxer que conté la llista de fitxers a processar.

  ```
  practica2 <nom_fitxer>
  ```

El codi té un pes d'un **80%** (codi amb funcions comentades, codi modular i net, ús correcte del llenguatge, bon estil de programació, el programa funciona correctament, tota la memòria és alliberada, sense accessos invàlids a memòria, etc.). El document tindrà un pes del **20%** restant (text ben estructurat, sense faltes d'ortografia i no passar-se del màxim nombre de pàgines).