Processamento de Linguagens (3º ano de Engenharia Informática)

Trabalho Prático Nº2

Relatório de Desenvolvimento

Ana Ribeiro	Carla Cruz	Jéssica Lemos
(a82474)	(a80564)	(a82061)

10 de Junho de 2019

Resumo

O trabalho que será apresentado tem como principais objetivos desenvolver *Processadores de Linguagens Regulares* que filtrem e transformem textos. Para tal efeito, utilizamos o sistema de produção para *filtragem de texto* GAWK.

Conteúdo

Capítulo 1

Introdução

De modo a desenvolver este segundo trabalho prático, foi necessário aplicar os conhecimentos adquiridos nas aulas acerca de expressões regulares e GAWK. O objetivo é desenvolver programas que processem a informação contida em textos preanotados com *Freeling*.
Assim, iremos apresentar as estratégias utilizadas, bem como a linha de pensamento seguida para resolver todos os exercícios propostos. Para além disso, serão explicados alguns pontos extras, que decidimos implementar para enriquecer o projeto desenvolvido.

Estrutura do Relatório

Inicialmente, no capítulo 2 é apresentada uma descrição detalhada do enunciado proposto para a realização deste projeto, bem como os objetivos deste.
De seguida, no capítulo 3 começamos por analisar a estrutura dos ficheiros fornecidos, de modo a tomar as decisões necessárias para responder a todas as questões do enunciado.
No capítulo 4 explicamos detalhadamente a implementação da solução dos problemas a dar resposta. Para além disto, apresentamos e especificamos algumas funcionalidades extra, que consideramos relevantes no âmbito do trabalho.
No capítulo 5, apresentamos os exemplos de utilização de todos os pontos do enunciado, bem como dos extras desenvolvidos.
Por fim, no capítulo 6 elaboramos uma análise crítica ao trabalho realizado.

Capítulo 2

Descrição do Problema

Este projeto tem como objetivo realizar a análise de ficheiros de texto. No nosso caso, estes encontram-se preanotados no formato *Freeling*. Para tal serão elaborados diversos programas que permitem responder às questões do enunciado, que se encontram apresentadas de seguida:

- Obter o número de extratos;
- Obter a lista dos nomes das personagens do Harry Potter e o respetivo número de ocorrências;
- Criar um ficheiro HTML para a lista dos verbos, substantivos, adjetivos e advérbios;
- Determinar o dicionário implícito na córpora.

Tendo em conta o enunciado decidimos elaborar pontos extras que consideramos úteis e que permitem enriquecer o trabalho desenvolvido, nomeadamente:

- Obter as estatísticas de um ficheiro com extratos;
- Criar uma ficheiro HTML com os tempos verbais, tipos de adjetivos e advérbios, com o respetivos número de ocorrências.

Capítulo 3

Textos preanotados com Freeling

Para desenvolver este trabalho foram disponibilizados cinco ficheiros. Estes vêm em formato *Freeling*, pelo que é possível obter as informações morfossintáticas das palavras presentes nos diversos textos. Cada ficheiro é constituído por diverso extratos, pelo que de seguida será apresentada a estrutura de um extrato.

```
1  Para         para         SP      SP  pos=adposition|type=preposition                                        - - (S:0(grup-sp:1(prep:1)                  - - -
-
2  a            o            DA0FS0  DA  pos=determiner|type=article|gen=feminine|num=singular                  - - (sn:3(espec-fs:2(j-fs:2))               - - -
-
3  Jessica      jessica      NP00000 NP  pos=noun|type=proper                                                   - - (grup-nom-fs:3(w-fs:3))))               - - -
-
4  que          que          PR0CN00 PR  pos=pronoun|type=relative|gen=common|num=invariable                    - - (prel:4)                                - - -
-
5  adora        adorar       VMIP3S0 VMI pos=verb|type=main|mood=indicative|tense=present|person=3|num=singular - - (grup-verb:5(verb:5))                - - -
-
6  histórias    história     NCFP000 NC  pos=noun|type=common|gen=feminine|num=plural                           - - (sn:6(grup-nom-fp:6(n-fp:6)))           - - -
-
7  Para_a_Ana   para_a_ana   NP00000 NP  pos=noun|type=proper                                                   - - (sn:7(grup-nom-ms:7(w-ms:7)))           - - -
-
8  que          que          PR0CN00 PR  pos=pronoun|type=relative|gen=common|num=invariable                    - - (prel:8)                                - - -
-
9  também       também       RG      RG  pos=adverb|type=general                                               - - (sadv:9)                                - - -
-
10 as           o            PP3FPA0 PP  pos=pronoun|type=personal|person=3|gen=feminine|num=plural|case=accusative - - (grup-verb:11(patons:10(paton-fp:10))  - - -
-
11 adora        adorar       VMIP3S0 VMI pos=verb|type=main|mood=indicative|tense=present|person=3|num=singular - - (grup-verb:11(verb:11)))               - - -
-
12 e            e            CC      CC  pos=conjunction|type=coordinating                                     - - (coord:12)                              - - -
-
13 para         para         SP      SP  pos=adposition|type=preposition                                        - - (grup-sp:13(prep:13)                    - - -
-
14 Di           di           NP00000 NP  pos=noun|type=proper                                                   - - (sn:14(grup-nom-ms:14(w-ms:14))))       - - -
-
15 ,            ,            Fc      Fc  pos=punctuation|type=comma                                             - - -                                       - - -
-
16 que          que          PR0CN00 PR  pos=pronoun|type=relative|gen=common|num=invariable                    - - (prel:16)                               - - -
-
17 ouviu        ouvir        VMIS3S0 VMI pos=verb|type=main|mood=indicative|tense=past|person=3|num=singular   - - (grup-verb:17(verb:17))               - - -
```

Figura 3.1: Excerto ficheiro harrypotter1

Neste as colunas encontram-se separadas por espaços. Para a realização deste projeto consideramos que a primeira coluna conterá o número da palavra, a seguinte a própria palavra, seguida do lema, pos e tag. Por fim, será possível observar as colunas *part of speech* e features.

Capítulo 4

Implementação da solução

4.1 Contar o número de extratos

Neste tópico pretende-se obter o número de extratos contidos no ficheiro de texto. Tendo em conta que cada extrato se encontra separado por uma linha em branco, consideramos que o número de extratos é igual à quantidade de linhas em branco. Como este tipo de linhas não possui colunas, o *number of fields*, NF, destas é 0. Deste modo, recorremos a uma variável *inc* que é incrementada sempre que é encontrada uma linha em branco. Quando o ficheiro chega ao fim, é impresso para o stdout o valor dessa variável, que corresponde ao número de extratos existentes, tal como pode ser observado de seguida:

```
BEGIN {FS=" "; inc=0}
NF==0 {inc++}
END {
      printf("Número de extratos: %i\n\n", inc);
}
```

Neste ponto decidimos acrescentar um extra, em que devolvemos a percentagem de cada um dos *part of speech*(*pos*) existentes em todos os extratos do ficheiro. Assim, sempre que para cada linha com informação, o *number o fields* é maior que zero, é guardado num array o número de vezes que aquele *pos* ocorre, para que quando for atingido o EOF do ficheiro, se possa calcular a percentagem de cada um e imprimir para o stdout essa informação. Neste array, os índices correspondem aos diferentes tipos de *part of speech* e o seu valor ao número de ocorrências de cada um. É importante referir que a informação acerca do *pos* se encontra no primeiro campo da sexta coluna de cada linha do extrato, pelo que foi necessário recorrer à função *split* para o obter, como se pode verificar em baixo:

```
BEGIN {FS=" "; inc=0}
NF==0 {inc++}
NF>0 {
      split($6,pos,"|");

      info[pos[1]] += 1;
      n++;
}

END {
      printf("Número de extratos: %i\n\n", inc);
```

```
      printf("============ Estatisticas ============\n");
      for (i in info)
            if (info[i] != 0)
                  printf("\t%s -> %3.2f %\n", i, info[i] *100/n);
}
```

4.2 Lista dos personagens do Harry Potter

Para elaborar a listagem de personagens, começou-se por guardar os nomes próprios destas numa estrutura, criando assim o array *nomes* que irá guardar o número de ocorrências do mesmo. Os índices deste array são os respetivos nomes encontrados. É possível verificar se estamos perante um nome próprio através do conteúdo da coluna seis ($6) do ficheiro. Desta forma, quando nesta coluna aparece a informação *"pos=noun | type=proper"* trata-se de um nome próprio. Na coluna dois ($2) é possível obter o nome, sendo este armazenado ou incrementado o número de ocorrências caso já exista no array.
Por fim, quando estamos perante a situação de EOF, o array armazenado será apresentado sobre a forma de lista ordenada. Para isto é utilizado *"sort -n -r"*, que terá a funcionalidade de ordenar o array *nomes*. Esta informação será impressa através da utilização de um ciclo *for* que percorrerá todos os elementos deste array, apresentando o nome e o número respetivo de ocorrências. Em conjunto, será apresentado o número total de personagens do Harry Potter, que será o tamanho do array *nomes*. Assim, de seguida, será apresentado a implementação da solução deste problema.

```
BEGIN {FS=" "; n=0}
NF>0 {
    if($6=="pos=noun|type=proper")
        nomes[$2]++;
}
END{
    for(i in nomes){print nomes[i], i | "sort -n -r"; n++}
    printf("Número total de personagens: %d\n\n", n);
}
```

4.3 Lista de Verbos, Substantivos, Adjetivos e Advérbios

Para criar as listas de verbos, substantivos, adjetivos e advérbios, foi utilizado um raciocínio semelhante ao anterior. Assim, foram criados quatro arrays, nomeadamente, *verb*, *subs*, *adj* e *adv*, dado que são necessárias quatro listagens.
Estas informações serão distinguidas através do conteúdo da coluna seis ($6), no campo *part of speech*, que indica se estamos perante um verbo, um substantivo, um adjetivo ou um advérbio para que este seja armazenado. Por exemplo, no caso do verbo, na informação desta coluna terá *pos=verb*. Para além de nos indicar que se trata de um verbo, esta pode conter mais informação, como por exemplo, os tempos verbais. Assim, é necessário adaptar a expressão regular utilizada e acrescentar os caracteres ".+". Nos arrays será armazenado o conteúdo da coluna dois ($2). Caso este já exista, irá ser incrementado o número de ocorrências do mesmo. Estas palavras são guardadas com as letras todas minúsculas de modo a ser possível a ordenação alfabética.
Quando é atingida a situação de EOF, são criados os diferentes ficheiros HTML para fazer as diferentes listagens pretendidas. Numa fase inicial é configurado o ficheiro de modo a ficar esteticamente agradável. De seguida, os arrays são ordenados alfabeticamente e cada um é percorrido com um ciclo *for*. Dentro deste

ciclo, é preenchida uma tabela com a informação dessa listagem. Por fim, esta tabela é impressa no ficheiro HTML. Em baixo, é possível observar o código implementado para a realização das listagens pretendidas:

```
BEGIN {FS=" "}
$6 ~/pos=verb.+/ {verb[tolower($2)]++;}
$6 ~/pos=noun.+/ {subs[tolower($2)]++;}
$6 ~/pos=adjective.+/ {adj[tolower($2)]++;}
$6 ~/pos=adposition.+/ {adv[tolower($2)]++;}

END{
    n = asorti(verb, ord)
    print "<html>\n<meta charset=\"UTF-8\"><h1>Verbos</h1>\n<table border=1>" > "verbos.html";
    for(i in ord){
       print "<tr><td width=50%>",ord[i],"</td><td>",verb[ord[i]],"</td><tr>" >"verbos.html";
    }

    // criação da tabela com a informação relativa aos substantivos

    // criação da tabela com a informação relativa aos adjetivos

    // criação da tabela com a informação relativa aos advérbios
}
```

4.4 Determinar o dicionário implicito no córpora

Neste ponto é expectável que seja apresentada uma lista com o *pos*, *lema* e a palavra associada. Para tal, começamos por criar um array no qual guardamos o *pos*, o *lema*, a palavra derivada associada bem como o número de vezes que esta ocorre. De modo a não existirem problemas na contagem de ocorrências das palavras e lemas, optámos por colocar estas em minúsculas, recorrendo à *tolower*. Tendo em conta que na coluna seis ($6) se encontra o *pos*, para cada um destes iremos armazenar o *lema*, cuja informação se encontra na coluna três ($3), e a respetiva palavra associada que se encontra na coluna anterior ($2). De seguida, para mostrar o resultado optámos por apresentar primeiro o *pos*, e posteriormente todos os respetivos *lemas*, as correspondentes palavras derivadas bem como o número de ocorrências destas. Para tal, quando é atingido o EOF do ficheiro percorremos o array através do auxílio de ciclos aninhados de modo a fazer-se a listagem. Assim, apresentamos um excerto do código elaborado para este ponto:

```
BEGIN { FS=" " }

NF>0 {
    results[$6][tolower($3)][tolower($2)]++;
}

END {
    for(i in results){
        printf("====== %s ======\n", i);
        for(j in results[i])
            for(k in results[i][j]){
                printf("\t%s -> %s => Nr Ocorrencias: %s\n", j, k, results[i][j][k]);
```

```
            }
        printf("\n");
    }
```

4.5 Informações das ocorrências

Tendo em conta o problema apresentado consideramos relevante apresentar uma página HTML com três tabelas, uma que corresponde aos tempos verbais com as correspondentes ocorrências, outra com os tipos dos substantivos e adjetivos e as suas ocorrências. Para tal, tivemos em conta o *pos* que se encontra na coluna seis ($6) do documento de modo a verificarmos se estamos perante um verbo, substantivo ou adjetivo. De modo a armazenar a informação de cada um criamos três arrays. Para acedermos ao campo *tense*, no caso de ser um verbo, ou *type* para os substantivos e advérbios, recorremos ao *split*. Sempre que este campo se repete é incrementado o número de ocorrências. Quando é atingido o EOF do ficheiro é criado o ficheiro HTML com as tabelas que contêm a informação pretendida como podemos observar na Figura 5.8. Cada tabela é preenchida recorrendo a um ciclo *for* que percorre o array pretendido. Apresentamos, assim, de seguida a forma como implementamos o programa:

```
BEGIN {FS=" "}
$6 ~/pos=verb.+/ {
    split($6,pos,"|");
    for(i in pos)
        if(pos[i] ~ /tense=.+/){
            split(pos[i], p, "=");
            tempos[p[2]]++;
            break;
        }
}
$6 ~/pos=noun.+/ {
    split($6,pos,"|");
    for(i in pos)
        if(pos[i] ~ /type=.+/){
            split(pos[i], p, "=");
            tipos[p[2]]++;
            break;
        }
}
$6 ~/pos=adjective.+/ {
 split($6,pos,"|");
    for(i in pos)
        if(pos[i] ~ /type=.+/){
            split(pos[i], p, "=");
            types[p[2]]++;
            break;
        }
}
END{
    // criação do ficheiro ocorrencias.html
}
```

Capítulo 5

Exemplos de Utilização

De seguida iremos expor um exemplo para cada programa desenvolvido, de modo a demonstrar como o mesmo funciona.

⇒ **Número de Extratos e Estatísticas**

Neste programa é apresentado o número de extratos contidos no ficheiro, bem como a percentagem de ocorrências de cada um dos *pos*.

```
[(master)⚡ % gawk -f numExtratos.awk harrypotter1
Número de extratos: 5569

============ Estatisticas ============
        pos=punctuation -> 16.27 %
        pos=adposition -> 11.59 %
        pos=number -> 0.48 %
        pos=verb -> 17.85 %
        pos=date -> 0.06 %
        pos=adverb -> 6.39 %
        pos=determiner -> 13.11 %
        pos=pronoun -> 7.35 %
        pos=noun -> 18.99 %
        pos=conjunction -> 4.28 %
        pos=interjection -> 0.15 %
        pos=adjective -> 3.49 %
```

Figura 5.1: Número de Extratos e Estatísticas

⇒ **Lista das Personagens**

Neste são apresentados os nomes das personagens do Harry Potter (nomes próprios) bem como o número de ocorrências no ficheiro do mesmo.

```
(master)⚡ % gawk -f personagens.awk harrypotter1
Número total de personagens: 670

1143 Harry
400 Ron
337 Hagrid
241 Hermione
150 Snape
143 Dumbledore
122 Dudley
109 Malfoy
108 Neville
106 Vernon
100 Quirrell
96 Mc_Gonagall
88 Gryffindor
67 Hogwarts
57 Petúnia
55 Slytherin
54 Quidditch
51 Dursley
50 Potter
49 Wood
48 Filch
47 Dursleys
38 Voldemort
```

Figura 5.2: Lista das Personagens

⇒ **Lista dos Verbos, Substantivos, Adjetivos e Advérbios**

Este programa desenvolvido permite visualizar 4 ficheiros HTML distintos:

- Verbos
- Substantivos
- Adjetivos
- Advérbios

Estes contêm uma tabela ordenada alfabeticamente e com o número de ocorrências.

Verbos

abafada	1
abafado	4
abafando	1
abafar	2
abalada	1
abalado	1
abanando	1
abanava	2
abandonar	2
abandonaram	1
abandonavam	1
abandoná	1
abanou	4

Figura 5.3: Lista de Verbos - verbos.html

Substantivos

a	8
aaaaaaaahhhh	1
aaaaaaarch	1
aaargh	2
aba	1
abafador	1
aberta	2
aberto	1
abertura	2
abeto	2
abraço	3
absinto	2

Figura 5.4: Lista dos substantivos - substantivos.html

Adjetivos

abelhudo	1
aberta	14
abertas	2
aberto	3
abertos	4
absoluta	5
absoluto	2
abundante	1
abundantes	1
académico	1
acastanhados	1
acesa	1
acesas	3

Figura 5.5: Lista dos Adjetivos - adjetivos.html

Adverbios

a	1999
a_bordo	1
a_fim_de	4
a_par_de	1
a_partir_de	5
a_tempo_de	6
a_toda_a	15
acima_de	1
além_de	2
ao_lado_de	6
ao_pé_de	1
apesar_de	23
após	9
através_de	5
atrás_de	12

Figura 5.6: Lista dos Advérbios - adverbios.html

⇒ Dicionário implícito no córpora

Este programa lista os *pos*, com os respetivos *lemas* e as palavras associadas, bem como o número de ocorrências.

```
[(master)⚡ % gawk -f dicionario.awk harrypotter1
====== pos=adjective|type=possessive|gen=masculine|num=singular|possessorpers=1|possessornum=singular ======
        meu -> meu => Nr Ocorrencias: 1

====== pos=adverb|type=negative ======
        jamais -> jamais => Nr Ocorrencias: 1
        nada -> nada => Nr Ocorrencias: 1
        não -> não => Nr Ocorrencias: 1283

====== pos=adjective|type=qualificative|gen=common|num=invariable ======
        clic -> clic => Nr Ocorrencias: 2
        recorde -> recorde => Nr Ocorrencias: 1
        co -> co => Nr Ocorrencias: 1
        simples -> simples => Nr Ocorrencias: 6
        hein -> hein => Nr Ocorrencias: 9
        prestes -> prestes => Nr Ocorrencias: 5
        contra -> contra => Nr Ocorrencias: 1
        extra -> extra => Nr Ocorrencias: 1
        tenpin -> tenpin => Nr Ocorrencias: 1
        oyt -> oyt => Nr Ocorrencias: 2

====== pos=verb|type=main|mood=indicative|tense=future|person=1|num=singular ======
        saber -> saberei => Nr Ocorrencias: 1
        poder -> poderei => Nr Ocorrencias: 1
        ter -> terei => Nr Ocorrencias: 1
        incomodar -> incomodarei => Nr Ocorrencias: 1
        mentir -> mentirei => Nr Ocorrencias: 1
        dever -> deverei => Nr Ocorrencias: 1
        estar -> estarei => Nr Ocorrencias: 1
        dizer -> direi => Nr Ocorrencias: 1
        possuir -> possuirei => Nr Ocorrencias: 1
        passar -> passarei => Nr Ocorrencias: 1
```

Figura 5.7: Dicionário implícito no córpora

⇒ **Número de ocorrências de determinado tempo verbal e tipo dos substantivos e adjetivos**

Este programa, desenvolvido como extra, permite a criação de um ficheiro HTML que contém três tabelas:

- Verbos - contém os diferentes tempos verbais e o número de ocorrências de cada
- Substantivos - contém os tipos e respetivas ocorrências
- Adjetivos - apresenta os tipos e número de ocorrências

Verbos

Ocorrências	Tempo
3224	present
249	conditional
155	future
3427	imperfect
4469	past
354	plusquamperfect

Substantivos

Ocorrências	Tipo
14010	common
5308	proper

Adjetivos

Ocorrências	Tipo
14	possessive
182	ordinal
3358	qualificative

Figura 5.8: Número de ocorrências de cada tempo verbal e tipo dos substantivos e adjetivos - ocorrencias.html

Capítulo 6

Conclusão

A elaboração deste trabalho prático permitiu-nos consolidar a matéria lecionada tanto nas aulas teóricas como práticas relativamente a GAWK. Assim, foi possível percebermos como é o funcionamento desta ferramenta para além de praticar o uso das diversas expressões regulares que permitem obter as informações pretendidas dos documentos. Ao longo da elaboração dos programas a eficiência desta ferramenta destacou-se quer mesmo ao nível da implementação tanto como o desempenho da solução elaborada.
Tendo em conta o desenvolvimento do projeto, concluímos que é notório a nossa evolução na utilização desta ferramenta e das expressões regulares, o que nos permitiu realização de alguns extras relevantes para o exercício em questão.