Universidade do Minho

Trabalho Prático nº1 (Gawk)

Processador de CETEMPúblico

Mestrado Integrado em Engenharia Informática

Processamento de Linguagens

(2º Semestre - 2017/2018)

a70565 Bruno Arieira

a78895 Jorge Cruz

a78494 José Dias

25 de Março de 2018

Resumo

"O CETEMPúblico (**C**orpus de **E**xtractos de **T**extos **E**lectrónicos **M**CT/**Público**) é um corpus de aproximadamente 180 milhões de palavras em português europeu, criado pelo projecto Processamento computacional do português após a assinatura de um protocolo entre o Ministério da Ciência e da Tecnologia (MCT) português e o jornal PÚBLICO em Abril de 2000."

Este trabalho, realizado no âmbito da unidade curricular Processamento de Linguagens, tem como fundamento analisar o ficheiro de texto CetemPúblico, de acordo com os requisitos pedidos no enunciado. Para isso, foram usadas expressões regulares recorrendo a um sistema de produção para filtragem de texto GAWK.

Conteúdo

1 Introdução

Neste primeiro trabalho prático pretende-se aplicar os conceitos abordados na disciplina Processamento de Linguagens, relativamente ao sistema de produção GAWK, com principal objetivo de aumentar a nossa experiência em ambiente Linux, a capacidade em termos de escrever expressões regulares, assim como aplicá-las de forma a torná-las útil para a análise de um documento.

Para a escolha deste trabalho, foram apresentados cinco enunciados, os quais foram atribuidos, segundo a aplicação de uma fórmula ao número de aluno mais baixo, a cada grupo. Com a aplicação desta fórmula, concluímos que nos foi atribuído o primeiro trabalho prático, Processador de CETEMPúblico. Este CORPORA é destinado a todos quantos desenvolvem programas de processamento a língua portuguesa, contendo artigos referentes ao jornal diário Português, Público.

Segundo este enunciado, necessitamos de analisar um ficheiro `txt`, através de programas `awk`, respondendo aos requisitos pedidos, assim como a outras funcionalidades definidas por nós.

2 Análise do texto-fonte

Este enunciado tem como principal objetivo a análise de ficheiros CORPORA, ou seja, ficheiros que contêm informação adicional de anotação sintática. Para tal foi nos fornecido três textos fontes `txt` de tamanho variavel. Segundo o tema atribuído, os ficheiros Corpora, usa tags xml para as anotações, em que as colunas representam a informação morfossintática de cada palavra. Assim, as colunas estão estruturadas pela seguinte ordem:

- Palavra
- Id da Secção
- Semestre
- Lema
- Pos: *Part of Speech*
- Tempo verbal-modo
- Num-pessoa
- Género
- Árvore

De seguida apresentamos um excerto do texto-fonte como ilustração e suporte para a estratégia adoptada.

```
<ext n=1 sec=clt sem=92b>
<p par=ext1-clt-92b-1>
<s>
</s>
<t>
Um    clt   92b   um    DET_arti   0    S    M    >N   0    0    0
revivalismo clt 92b   revivalismo N    0    S    M    NPHR 0    0    0
refrescante clt 92b   refrescante ADJ  0    S    M    N<   0    0    0
</t>
</p>
<p par=ext1-clt-92b-2>
<s>
O     clt   92b   o     DET_artd   0    S    M    >N   0    0    0
7     clt   92b   7=e=meio   NUM_card   0    S    M    SUBJ> 0    0    0
e     clt   92b   7=e=meio   NUM_card   0    S    M    SUBJ> 0    0    0
Meio  clt   92b   7=e=meio   NUM_card   0    S    M    SUBJ> 0    0    0
é     clt   92b   ser   V     PR_IND     3S   0    FMV  0    0    0
um    clt   92b   um    DET_arti   0    S    M    >N   0    0    0
ex-libris   clt   92b   ex-libris   N    0    S    M    <SC  0    0    0
da    clt   92b   de+o  PRP+DET_artd    0    S    F    N<+>N 0    0    0
noite clt   92b   noite N    0    S    F    P<   0    0    0
algarvia    clt   92b   algarvio   ADJ  0    S    F    N<   0    0    0
.     clt   92b   .     PU    0    0    0    PONT 0    0    0
</s>
<s>
É     clt   92b   ser   V     PR_IND     3S   0    FMV  0    0    0
uma   clt   92b   uma   NUM_card   0    S    F    <SC  0    0    0
das   clt   92b   de+o  PRP+DET_artd    0    P    F    N<+>N 0    0    0
mais  clt   92b   mais  ADV_quant  0    0    0    >A   0    0    0
antigas     clt   92b   antigo     ADJ  0    P    F    >N   0    0    0
discotecas clt   92b   discoteca  N    0    P    F    P<   0    0    0
do    clt   92b   de+o  PRP+DET_artd    0    S    M    N<+>N 0    0    0
Algarve     clt   92b   Algarve    PROP 0    S    F    P<   0    0    0
```

Como se pode verificar, cada extrato contem o seu ID, parágrafos, frases, *multi-word-expressions*, etc. Mais detalhadamente, as tags são atribuídas a cada conceito do seguinte modo:

corpus	`<corpus> extracto+ </corpus>`
extracto	`id_extracto conteúdo_extracto </ext>`
contéudo_extracto	`parágrafo+`
parágrafo	`título | identif_autor | <p> frase+ </p> | elemento_lista`
título	`<t> token+ </t>`
identif_autor	`<a> token+ </a>`
elemento_lista	`<li> token+ </li>`
frase	`( <s> | <s tipo=frag> ) token+ </s>`
token	`<marca num= X > | palavra | sinal_pontuação | identificador`
id_extracto	`<ext n=número sec=id_sec sem=semestre >`
número	`[0-9]+`
id_sec	`soc | pol | clt | des | opi | eco | com | clt-soc | pol-soc | nd`
semestre	`91a | 91b | 92a | 92b | 93a | 93b | 94a | 94b | 95a | 95b | 96a | 96b |97a | 97b | 98a | 98b`

2.1 Ações Semânticas

Depois de uma análise detalhada do ficheiro `txt` a processar, deparamo-nos que em cada linha continha a maior parte da informação relevante para a realização do trabalho. Com isto, definimos:

- *Field Separator* (separador de campo): normalmente define o `\n` mas alterámos para `\t`, pois a informação relativamente ás palavras encontra-se estruturada por colunas divididas por um tab.

2.2 Modificações no texto-fonte

Após a análise do ficheiro foram encontradas tags com erros que tiveram de ser substituídas pela respetiva tag correta, de modo a evitar defeitos nas repostas produzidas. Ao longo deste relatório, todos os resultados que forem apresentados, foram obtidos a partir do ficheiro alterado, pelo que, poderá haver breves diferenças no *output* dos programas em relação ao texto original.

Estas tags erradas foram descobertas procurando todos as expressões diferentes que começassem no início das linhas com o carater `"<"`. Assim, a expressão regular correspondente foi: `^<\w+`.

3 Filtro de Texto - Sistema de Produção GAWK

Depois da análise e interpretação do ficheiro com a ajuda do enunciado fornecido, podemos entao proceder ao desenvolvimentos de filtros de textos utilizando o sistema de produção de filtragem de texto GAWK.

3.1 Números de Extratos, Parágrafos e Frases

Neste exercício, o principal objetivo é contar o número de extratos, parágrafos e frases. Para tal, foi necessário criar três contadores (`contaEXT`, `contaPAR`, `contaFRA`) inicializados a 0, onde sempre que aparece a tag relativa a cada um dos três parâmetros (`<ext/`, `<p/`, `<s/`), e o primeiro caratere da linha é `"<"`, adiciona uma ocorrência ao contador respetivo.

```
BEGIN          { contaEXT = 0; contaPAR = 0; contaFRA = 0; }
/^<ext/        { contaEXT++ }
/^<p/          { contaPAR++ }
/^<s/          { contaFRA++ }
END            {
                 print "Extratos = " contaEXT;
                 print "Paragrafos = " contaPAR;
                 print "Frases = " contaFRA;
               }
```

Comando e Output:

```
  $ awk -f numberOfExtParSen.awk Cetempublico01.txt
Extratos = 5455
Paragrafos = 12843
Frases = 29111
```

3.2 Lista das *multi-word-expressions* e respectivo número de ocorrências

Com o seguinte programa, é possível obter a lista de todas as *multi-word-expressions* (ignorando as letras maiúsculas e minúsculas) e o respetivo número de ocorrências das expressões. A resolução deste exercício tornou-se fácil através da função `tolower()`, que passa todas as letras maiúsculas de uma `string` para minúsculas. A partir daí, recorrendo a um array, bastou apenas guardar o número de vezes que uma expressão aparecia durante o ficheiro.

Importante referenciar que o programa apenas avalia as palavras que se encontram entre as tags das *multi-word-expressions*, pelo que foi necessário uma variável `var` que, quando o seu valor fosse 1, significava que essa linha teria de ser processada. O valor desta variável é então controlada a partir das tags referidas.

```
BEGIN          { FS = "\t" ; var = 0 ; string = "" }
/<\/mwe>/      { var = 0; words[tolower(string)]++ }
var == 1       { string = string" "$1 }
/<mwe /        { var = 1; string = "" }
END            { for( i in words) print i,"=", words[i] }
```

Comando e parte do Output:

```
 $ awk -f listMWE.awk Cetempublico01.txt
dois dedos = 1
além disso = 41
meio ambiente = 3
a bem = 4
taxas de câmbio = 6
ondas de choque = 1
de primeira = 8
estão nas mãos = 1
salto alto = 1
in vitro = 1
café da manhã = 1
passagem de nível = 2
por maioria = 6
escolas secundárias = 8
às avessas = 1
no fio = 1
fogos postos = 1
de perto = 13
assim como = 31
em criança = 2
nos termos de = 2
```

3.3 Lista dos verbos e respectivo número de ocorrências

Pretende-se calcular a lista de todas as palavras que são verbos no ficheiro. Para verificar que é um verbo, é necessário ir á coluna `pos` (`$5`), e determinar se `pos=V`. Caso seja, incrementa no array `verbs` o valor da posição do respetivo nome do verbo, que corresponde ao lema (`$4`). No fim, para cada posição do array, imprime-se o nome do verbo e o número de ocorrências.

```
BEGIN          { FS = "\t" }
$5 == "V"      { verbs[$4]++ }
END            { for (i in verbs) print i, "=", verbs[i] }
```

Comando e parte do Output:

```
  $ awk -f listOfVerbs.awk Cetempublico01.txt
apedrejar = 2
insitir = 1
assinar = 128
poder = 1491
comedir = 2
podar = 4
pipa = 1
ressuscitar = 3
dentada = 2
povoar = 5
tolerar = 6
recusar = 82
impingir = 1
erguer = 9
imprimir = 12
depender = 41
prometer = 84
ajustar = 7
recair = 18
processar = 13
receber = 294
```

3.4 Dicionário implícito no corpora

Esta funcionalidade permite filtrar as palavras com o lema e o *part of speech* correspondente a cada uma. Caso o primeiro caratere seja uma letra (`[a-zA-Z]`), o `print` é efetuado, imprimindo o lema (`$4`) e o pos (`$5`) correspondente. Para que seja ignorado os casos de letras maiúsculas e minúsculas, é necessário utilizar a função `tolower()` novamente. Deste modo, torna-se possível avaliar se uma palavra já foi imprimida verificando o valor da sua posição no array.

```
BEGIN                   { FS = "\t" }
/^[a-zA-Z]+/            { word = tolower($1) ;
                          if (array[word] == 0) {
                                array[word]++ ;
                                print $1"\t\t"$4"\t\t"$5
                                }
                        }
END                     { }
```

Comando e Output:

```
  $ awk -f dictionary.awk Cetempublico01.txt
um              um              DET_arti
revivalismo             revivalismo             N
refrescante             refrescante             ADJ
o               o               DET_artd
e               7=e=meio                NUM_card
meio            7=e=meio                NUM_card
ex-libris               ex-libris               N
da              de+o            PRP+DET_artd
noite           noite           N
algarvia                algarvio                ADJ
uma             uma             NUM_card
das             de+o            PRP+DET_artd
mais            mais            ADV_quant
antigas         antigo          ADJ
discotecas              discoteca               N
do              de+o            PRP+DET_artd
algarve         Algarve         PROP
situada         situar          V
em              em              PRP
albufeira               Albufeira               PROP
que             que             SPEC_rel_clb-fs
```

3.5 Funcionalidades Adicionais

3.5.1 Reconstrução do Texto Original

Esta funcionalidade extra permite reconstituir o texto utilizado para a criação do ficheiro CETEMPúblico. Para recriar o texto foi utilizada o campo `$1` dos vários registos tendo em conta inícios de parágrafos (`<p>`), títulos (`<t>`), expressões (`<mwe>`) e também pontuação e símbolos especiais.

A necessidade de ter em conta a pontuação surge pois ela requer uma alteração no espaçamento entre as próximas palavras ou expressões. Por exemplo, quando aparecem aspas é necessário que estas estejam ambas juntas às expressões que rodeiam. Através de uma variável `nospace` é possível avisar a próxima palavra para imprimir ou não um espaço atrás. No caso das aspas, é necessário existir outra variável (`quotes`) que indica à próxima aparição do símbolo para não imprimir espaço atrás. Obviamente, existem mais símbolos que não constam no programa, porém, todos os que o ficheiro CETEMPúblico contém, estão assinalados no programa.

```
BEGIN                                { FS = "\t" ; quotes = 0 ; nospace = 0 }

/^[^[:punct:]]/                      { if ( nospace ) { nospace = 0 ; printf("%s", $1) }
                                       else printf(" %s", $1)
                                     }

/^\"/                                { if ( quotes ) printf("%s", $1)
                                       else { quotes = 1 ; nospace = 1 ; printf(" %s", $1) }
                                     }

/^[\!|\%|\'|\)|\,|\.|\:|\;|\?|\»]/  { printf("%s", $1) }
/^[\(|\`|\«]/                        { nospace = 1 ; printf(" %s", $1) }
/^[\&|\+|\-|\=]/                     { printf(" %s", $1) }
/^\//                                { printf("%s", $1) ; nospace = 1 }

/^(<t)/                              { printf("\n") }
/^(<p)/                              { printf("\n\t") }
/^(<li>)/                            { printf("\n\t->") }
/^(<a>)/                             { printf("\n") }

END                                  { }
```

Comando:

```
$ awk -f convertToTxt.awk Cetempublico01.txt > text.txt
```

Parte do Output (text.txt):

```
 Um revivalismo refrescante
      O 7 e Meio é um ex-libris da noite algarvia. Éuma das mais antigas discotecas do
Algarve, situada em Albufeira, que continua a manter os traços decorativos e as clientelas
de sempre. É um pouco a versão de uma espécie de «outro lado» da noite, a meio caminho entre
os devaneios de uma fauna periférica, seja de Lisboa, Londres, Dublin ou Faro e Portimão, e
a postura circunspecta dos fiéis da casa, que dela esperam a música «geracionista» dos 60 ou
dos 70. Não deixa de ser, nos tempos que correm, um certo «very typical» algarvio, cabeça de
cartaz para os que querem fugir a algumas movimentações nocturnas já a caminho da
ritualização de massas, do género «vamos todos ao Calypso e encontramo-nos na Locomia».
```

3.5.2 Conversão de Ficheiro para HTML

Do mesmo modo que o ficheiro foi convertido anteriormente para o texto original, também agora foi possível construir um ficheiro em HTML, com a mesma informação. A única diferença encontra-se no facto de ser necessário acrescentar as tags corretas a cada seccção.

```
BEGIN                                { FS = "\t" ; quotes = 0 ; nospace = 0 ;
                                       print "<html>" ;
                                       print "<head>" ;
                                       print "<meta charset=\"utf-8\">" ;
                                       print "<title>"FILENAME"</title>" ;
                                       print "</head>" ;
                                       print "<body>"
                                     }
/^(<\/t)/                            { printf("</h3>\n") }
/^(<\/p)/                            { printf("</p>\n") }
/^(<\/li>)/                          { printf("</li></ul>\n") }
/^(<\/a>)/                           { printf("</address>\n") }
/^[^[:punct:]]/                      { if ( nospace ) { nospace = 0 ; printf("%s", $1) }
                                       else printf(" %s", $1)
                                     }
/^\"/                                { if ( quotes ) printf("%s", $1)
                                       else { quotes = 1 ; nospace = 1 ; printf(" %s", $1) }
                                     }
/^[\!|\%|\'|\)|\,|\.|\:|\;|\?|\»]/   { printf("%s", $1) }
/^[\(|\`|\«]/                        { nospace = 1 ; printf(" %s", $1) }
/^[\&|\+|\-|\=]/                     { printf(" %s", $1) }
/^\//                                { printf("%s", $1) ; nospace = 1 }
/^<t/                                { printf("\n<h3>") }
/^<p/                                { printf("\n<p>") }
/^(<li>)/                            { printf("\n<ul><li>") }
/^(<a>)/                             { printf("\n<address>") }
END                                  { print "</body>" ;
                                       print "</html>" }
```

Comando:

```
$ awk -f convertoToHtml.awk Cetempublico01.txt > text.html
```

Parte do Output (text.html):

```
<html>
<head>
<meta charset="utf-8">
<title></title>
</head>
<body>

<p>
<h3> Um revivalismo refrescante</h3>
</p>

<p> O 7 e Meio éum ex-libris da noite algarvia. Éuma das mais antigas discotecas do Algarve
,situada em Albufeira, que continua a manter os traços decorativos e as clientelas de sempre
```

3.5.3 Lista de extratos relativos a uma palavra

Esta funcionalidade tem por objetivo dar os respetivos extratos, que contenham uma palavra na sua informação, passada como argumento. Uma das maneiras de fazer este programa foi mudar o RS (*record separator*) para "<ext ", ou seja, cada registo representava todo o conteúdo do extrato. Assim sendo, bastou apenas procurar a palavra pretendida em todo o registo ($0) e, caso esta fosse encontrada, imprimir o número do extrato que corresponde ao primeiro campo do registo ($1).

A palavra que se pretende procurar é passada ao programa através de uma variável var. Para apenas avaliar as palavras que se encontram no início das linhas, é importante adicionar "\n"antes da palavra passada como parâmetro. Esta tarefa deve-se ao facto de que o FS (*field separator*) não ser um "\t"como nos casos anteriores, pois é necessário que seja um espaço (predefinido, não sendo necessário assinalar no programa) para poder imprimir o número do extrato em questão.

```
BEGIN                 { RS = "<ext " ; expression = "\n"var }
$0 ~ expression       { print $1 }
END                   { }
```

De seguida, podemos ver um exemplo de pesquisar a palavra "Portimão"e o resultado obtido com o número dos extratos que apresentam esta palavra no seu texto.

Comando e Output:

```
  $ awk -v var="Portimão" -f searchWord.awk Cetempublico01.txt
n=1
n=1040
n=1166
n=1649
n=2532
n=3190
n=3460
n=3472
n=4224
n=4537
n=5247
n=5257
```

4 Conclusão

Com este trabalho consolidamos os conceitos aprendidos das aulas da melhor forma, pois permitiu nos obter maior experiência a implementar o sistema de produção para filtragem de texto GAWK. Concluímos que este sistema tem um predomínio enorme de processamento, pois através de poucas linhas de código, somos capazes de processar grandes quantidades de informação.

Tendo em conta o enunciado proposto, conseguimos efetuar todas as funcionalidades propostas e ainda fomos capazes de adicionar três funcionalidades extras, da reconstituição do texto inicial, da conversão para um ficheiro html e a lista de extratos relativos a uma palavra.

Referências

https://catalog.ldc.upenn.edu/docs/LDC2001T62/info_pt.html