Universidade do Minho

Mestrado Integrado em Engenharia Informática

Departamento de Informática

Processamento de Linguagens

Dicionário Cinematográfico

António Sérgio Alves Costa A78296
Isabel Sofia da Costa Pereira A76550
Maria de La Salete Dias Teixeira A75281

12 de Junho de 2018

Conteúdo

1 Introdução

No âmbito da unidade curricular de Processamento de Linguagens, foi-nos proposto o desenvolvimento de um Sistema de Tratamento de Textos Cinematográficos, isto é, de um programa capaz de processar textos cinematográficos tendo por base um dicionário cinematográfico criado pelo grupo. Assim, o objetivo deste trabalho é, partindo de um texto, identificar palavras presentes no dicionário cinematográfico, adicionar um *footnote* com o termo da palavra em inglês e, no final do respetivo texto, criar um apêndice com as informações das palavras encontradas.

No desenvolvimento do dicionário cinematográfico, teve-se em conta para cada palavra o seu significado, variações, designação comum em inglês e sinónimos.

Para a realização do programa, recorreu-se às ferramentas *flex* e *yacc* e à linguagem de programação C.

2 Descrição do Problema

Ao analisar o problema foi possível observar que este poderia ser dividido em três partes.

Primeiro, seria necessário o desenvolvimento de um dicionário cinematográfico que contivesse os paramêtros palavra, significado, variações, designação em inglês e sinónimos.

Segundo, seria preciso o desenvolvimento de um programa que fizesse o parsing do dicionário guardando toda a sua informação numa estrutura.

Terceiro, e com a estrutura preenchida, seria fundamental a implementação de um programa que lêsse um texto e identificasse as palavras cinematoráficas presentes neste aplicando sobre estas a transformação requerida.

Com todos estes passos poderia-se então desenvolver-se o Sistema de Tratamento de Textos Cinematográficos, como se pode verificar pela figura seguinte.

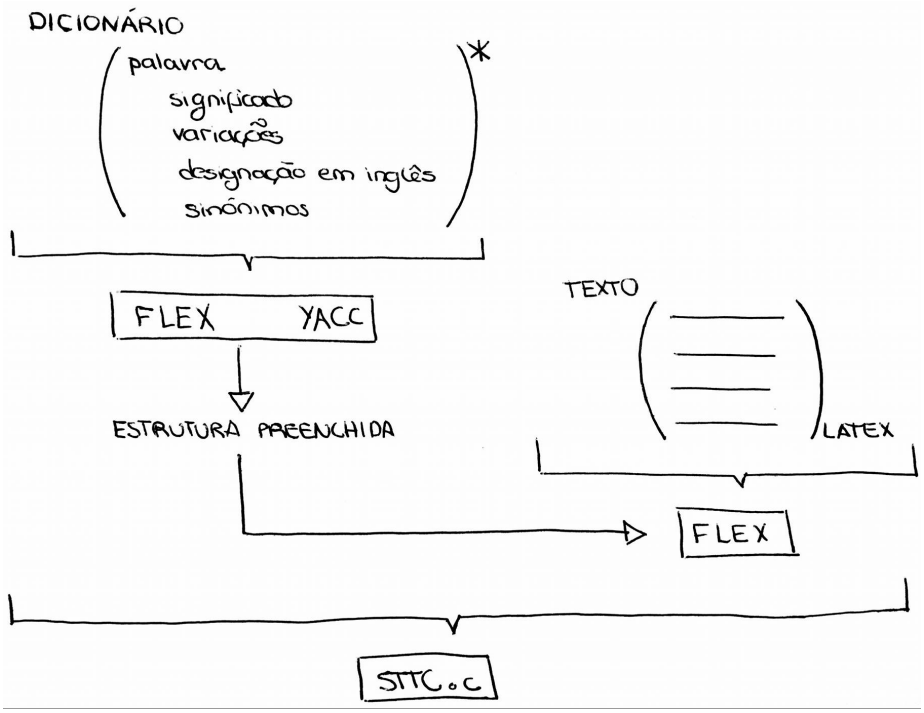

Figura 1: Esquema do projeto

3 Implementação

Para implementar o programa, foi necessário proceder ao desenvolvimento de uma estrutura para armazenar os dados, de um ficheiro *flex* e de um ficheiro *yacc* para a parte do dicionário cinematográfico. De seguida, foi necessário desenvolver um ficheiro *flex* para o texto cinematográfico. A estrutura para armazenamento de dados desenvolvida é uma tabela de *hash*, completamente implementada pelo grupo.

Para se conseguir ligar todos os ficheiros, foi necessário criar um ficheiro .c, sendo este o único que contém uma *main*. Deste modo, este é o ficheiro principal, sendo este o que gera o executável do programa.

Para além disso, como se está na presença de dois ficheiros *flex*, foi necessário alterar a extensão de um deles. Como o *flex* do dicionário está associado a um *yacc*, verificou-se que alterar a extensão deste geraria problemas no *yacc*. Então, alterou-se o *flex* do texto, que em vez de gerar um *lex.yy.c* resulta em *lex.texto.c*.

Por motivos de conveniência, criou-se uma *Makefile* que faz todas as compilações necessárias.

3.1 Dicionário Desenvolvido

O dicionário cinematográfico foi desenvolvido em formato tabelar, recorrendo ao HTML. O uso do HTML é justificado por motivos de estética e pela simplicidade da linguagem.

Sendo que foram tomados em conta quatro argumentos para cada palavra, a tabela é constituída por cinco colunas, sendo estas:

- palavra;
- significado;
- variações;
- designação comum em inglês;
- sinónimos.

A tabela foi desenvolvida recorrendo ao CSS. Assim, as entradas do dicionário estão no formato do código apresentado de seguida, que por sua vez corresponde à geração da tabela da figura 2.

```
<tr class="bg-primary">
  <td name="palavra">ator</td>
  <td name="significado">Pessoa que interpreta um papel,
  ↪  encarnando uma personagem.</td>
  <td name="variacoes">atriz; atores; atrizes</td>
  <td name="ingles">actor</td>
  <td name="sinonimos">-</td>
</tr>
```

Dicionário Cinematográfico

Palavra	Significado	Variações	Designação em Inglês	Sinonimos
ator	Pessoa que interpreta um papel, encarnando uma personagem.	atriz; atores; atrizes	actor	-
banda sonora	Gravação musical utilizada num filme.	-	soundtrack	-
cena	A própria ação ou representação teatral.	cenas	scene	ação; acontecimento
cenário	Conjunto dos bastidores e vistas apropriadas aos factos que se representam; lugar onde decorre um acontecimento, uma ação.	cenários	scenario	plano de fundo; paisagem
clímax	Momento determinante na ação a partir do qual se gera o desfecho ou esclarecimento dos acontecimentos.	-	climax	auge; momento culminante
corte	Interrupção do registo pela camêra	-	cut	interrupção
créditos	Lista dos nomes dos profissionais e entidades envolvidos direta ou indiretamente num trabalho (filme, documentário, etc.); ficha técnica.	-	credits	-

Figura 2: Dicionário Cinematográfico

3.2 Processamento do Dicionário

Para processar e armazenar o dicionário desenvolvido, foi necessário criar uma estrutura capaz de o suportar e, por motivos de eficiência, de rápida procura. Assim, foi desenvolvida uma tabela de *hash*. Para além disso, foi necessário desenvolver um reconhecedor léxico e sintático, tendo-se criado um ficheiro *flex*, e uma gramática, tendo-se criado um ficheiro *yacc*.

3.2.1 Armazenamento dos Dados

Tal como mencionado anteriormente, pretendeu-se utilizar uma estrutura de procura rápida e capaz de armazenar as várias componentes do dicionário. Para tal, considerou-se uma *hash* dinâmica a estrutura mais apropriada.

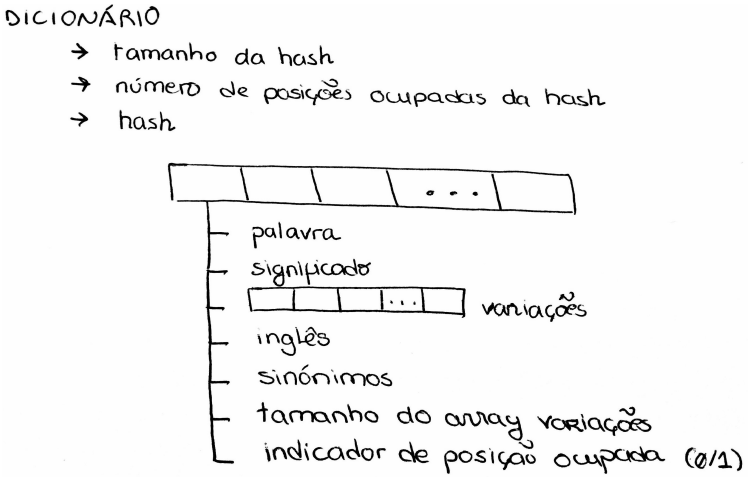

Figura 3: Estrutura para armazenamento dos dados do dicionário

Com a estrutura apresentada na figura 3 desenvolveu-se então o ficheiro *hash.c* e *hash.h* com a definição da estrutura e com os métodos cruciais para a manipulação da mesma. Sendo assim, criaram-se os segintes métodos:

- **init**, responsável por iniciar a estrutura;
- **hash**, retorna a posição na *hash* consoante o código de uma determinada palavra;
- **copiar**, insere informação numa certa posição da *hash*;
- **insert**, verifica se a *hash* já se encontra com metade das posições ocupadas e adiciona informação, com o auxilio da função **copiar**. Caso a *hash* se encontre cheia, a função cria uma nova com o dobro do tamanho e com a mesma informação da antiga;
- **exists**, verifica se uma palavra, que pode também ser uma variação, existe na *hash*;
- **allLower**, colaca as letras de uma palavra em minúsculas para uma comparação válida entre essa palavra e as palavras do dicionário;
- **printInfo**, imprime a informação de uma palavra em formato tabelar. Utilizada para a elaboração do apêndice;
- **getPalavra**, devolve a palavra principal do dicionário associada a uma determinada palavra, porque esta pode ser uma variação;
- **getIngles**, retorna a designação em inglês da palavra principal do dicionário associada a uma determinada palavra, porque esta pode ser uma variação;
- **clean**, limpa a estrutura.

Para além disso, o ficheiro *hash.h* serve como ponte para que os vários ficheiros *flex*, *yacc* e *.c* possam utilizar a estrutura.

3.2.2 Ficheiro *Flex*

Para processar o dicionário foi necessário especificar duas coisas: as expressões regulares que o definem e a ação a executar em cada uma destas. Tendo em conta que cada entrada no dicionário é constituída por palavra, significado, variações, designação em inglês e sinónimos, foi criada uma expressão regular para cada caraterística, de maneira a se obter a informação correta para cada elemento da estrutura de dados.

Como a linguagem utilizada no dicionário foi HTML, as expressões regulares criadas são bastante simples, tendo que se procurar apenas pela etiqueta, *name*, a que se refere a linha, tal como se pode verificar no excerto de código em baixo.

Para cada *match* feito, a ação a executar é bastante semelhante, mudando apenas o *token* a retornar. Como é essencial distinguir as cinco características

de cada entrada, utilizou-se cinco *tokens*, cada um referente a uma caraterística, para posteriormente se poder implementar a gramática definida no *yacc*. Antes de o *token* ser retornado, iguala-se o *yyval.tex* ao *yytext*, sendo que o valor *text* está definido no *yacc* como um *char**.

Assim, o analisador léxico desenvolvido para o dicionário foi o seguinte.

```
[ \t\n]                                     ;
\<td" "name=\"palavra\"\>.*\<\/td\>         { yytext[yyleng-5]='\0';
                                              yylval.text = yytext+19;
                                              return PALAVRA;
                                            }
\<td" "name=\"significado\"\>.*\<\/td\>     { yytext[yyleng-5]='\0';
                                              yylval.text = yytext+23;
                                              return SIGNIFICADO;
                                            }
\<td" "name=\"variacoes\"\>.*<\/td\>        { yytext[yyleng-5]='\0';
                                              yylval.text = yytext+21;
                                              return VARIACOES;
                                            }
\<td" "name=\"ingles\"\>.*\<\/td\>          { yytext[yyleng-5]='\0';
                                              yylval.text = yytext+18;
                                              return INGLES;
                                            }
\<td" "name=\"sinonimos\"\>.*\<\/td\>       { yytext[yyleng-5]='\0';
                                              yylval.text = yytext+21;
                                              return SINONIMOS;
                                            }
.                                           ;
```

3.2.3 Ficheiro *yacc*

Para desenvolver o *yacc* do dicionario, começou-se por desenvolver a gramática para o mesmo. A gramática é constituída por uma sequência de comandos, em que cada comando pode ser uma palavra, significado, variações, inglês (referente à designação em inglês) ou sinónimos. Assim, criou-se a seguinte gramática:

```
SeqComandos : SeqComandos Comando
            |
            ;

Comando : PALAVRA
        | SIGNIFICADO
        | VARIACOES
        | INGLES
        | SINONIMOS
        ;
```

Os *tokens* criados correspondem aos apresentados no Comando, ou seja, a PALAVRA, SIGNIFICADO, VARIACOES, INGLES e SINONIMOS. A necessidade de criar estes *tokens* centra-se no facto de ser essencial distinguir a característica que se está a processar. Caso contrário, a informação inserida na estrutura de dados poderia não ser a correta, por exemplo, nos sinónimos poderia corresponder à designação em inglês.

Sendo que o tipo da informação é todo o mesmo, isto é, todas as características correspondem a *char**, apenas se criou na *union* um *char* text*, tal como já se referiu no tópico anterior.

Relativamente à ação a executar, esta é bastante simples, sendo apenas necessário guardar a informação presente no $1 numa variável para depois esta ser inserida no dicionário. A informação total é adicionada no dicionário quando são encontrados os sinónimos, pois esta é a última informação de cada entrada. Assim, no SINONIMO é evocada a função **insert** da estrutura de dados.

Desta forma, a gramática implementada no *yacc* foi a seguinte:

```
SeqComandos : SeqComandos Comando
            |
            ;

Comando : PALAVRA       { palavra = (char*) malloc(strlen($1)*sizeof(char));
                          strcpy(palavra,$1);}
        | SIGNIFICADO   { significado = (char*) malloc(strlen($1)*sizeof(char));
                          strcpy(significado,$1);}
        | VARIACOES     { variacoes = (char*) malloc(strlen($1)*sizeof(char));
                          strcpy(variacoes,$1);}
        | INGLES        { ingles = (char*) malloc(strlen($1)*sizeof(char));
                          strcpy(ingles,$1);}
        | SINONIMOS     { sinonimos = (char*) malloc(strlen($1)*sizeof(char));
                          strcpy(sinonimos,$1);
                          dicionario = insert(dicionario,palavra,
                            significado,variacoes,ingles,sinonimos);
                        }
                        ;
```

Para se ter acesso à estrutura de dados criada no programa principal e para realizar o *yyparse()*, foi necessário desenvolver uma função que recebe a estrutura de dados inicializada e o nome do ficheiro HTML, onde se encontra o dicionário cinematográfico a ser processado. Assim, criou-se a função **load** que retorna a estrutura preenchida.

```
Dic load(Dic dic, char* filename){
    dicionario = dic;
    yyin = fopen(filename, "r");
    yyparse();
    return dicionario;
}
```

Por motivos de execução, foi necessário criar um *header* para que os outros ficheiros reconheçam a função load, dando origem ao ficheiro *lexDicionario.h*.

3.3 Processamento dos Ficheiros de Texto a Anotar

Para transformar os textos cinematográficos, foi necessário desenvolver um analisador léxico capaz de processar a informação presente no texto e que, cada vez que fossem encontradas palavras presentes no dicionário, ou variações das mesmas, adicionasse uma *footnote* e armazenasse a palavra de maneira a esta ser incluída no apêndice final. Para isto, foi desenvolvido um ficheiro *flex*.

3.3.1 Ficheiro *Flex*

Para se desenvolver um bom analisador léxico para o texto foi necessário ter em consideração o dicionário. Após um período de análise do dicionário, notou-se que este pode conter entradas com duas palavras, ou seja, é necessário não só analisar o texto palavra a palavra, como também analisar conjuntos de duas palavras.

Assim sendo, começou-se por dividir o texto em palavras para que fosse possível compará-las às entradas do dicionário. Com os conjuntos de duas palavras, decidiu-se dar prioridade a estes. Por exemplo, vamos imaginar que temos o excerto de frase "o ator secundário", neste exemplo iriamos analisar primeiro o conjunto de palavras "o ator", de seguida iriamos analisar "o", após isto "ator secundário"e assim sucessivamente.

Além disto, como se implementou as variações de cada palavra, isto implica comparar a palavra, ou conjunto de palavras, a todas as variações de cada entrada do dicionário. Esta comparação é feita através da função **exists** da *hash.c*.

De seguida, apresenta-se uma simplificação do código implementado para esta funcionalidade.

```
if(existe no dicionario as 2 palavras juntas){
        printf("\\textbf{%s}\\footnote{%s}%s", juntas,
         ↪  getIngles(dicionario,juntas), pontAtual);

        existef(); //verifica se a entrada já apareceu no texto
        if(!existe){
                naoexiste(); //armazena a entrada
        }
}
else{
        if(existe no dicionario a palavra *antiga*){
                printf("\\textbf{%s}\\footnote{%s}%s", antiga,
                 ↪  getIngles(dicionario,antiga), pontAnt);
```

```
                existef(); //verifica se a entrada já apareceu
                if(!existe){
                        naoexiste(); //armazena a entrada
                }
        }
        else{ //não é palavra cinematográfica pelos dois modos
                printf("%s", antiga);
                printf("%s", pontAnt);
        }

        antiga = atual;
        pontAnt = pontAtual;
}
```

Para além das variações, decidiu-se criar mais um extra que faria o analisador léxico não processar texto que tivesse dentro do comando *verbatim*.

Para a utilização desta função foi necessário criar o contexto *verb*. Este contexto é iniciado sempre que se encontra o comando \begin{verbatim}. Após isso, apenas é necessário imprimir, sem qualquer alteração, todo o texto encontrado até o comando \end{verbatim}, comando que faz terminar o contexto *verb*.

```
^\\begin\{verbatim\}        { printf("\\begin{verbatim}");
                              BEGIN verb;
                            }

<verb>^\\end\{verbatim\}    { printf("\\end\{verbatim\}");
                              BEGIN 0;
                            }

<verb>.*                    { printf("%s",yytext);}
```

Por fim, quando se atinge o \end{document}, se foram encontradas palavras cinematográficas, imprimi-se estas no apêndice em formato tabelar, caso contrário esse processo não se realiza. De seguida, imprime-se então o comando encontrado.

```
if(tamArray){
  printf("\\clearpage\n");
  printf("\\begin{appendix}\n");
  printf("\\section{Dicionário Cinematográfico}\n");
  printf("\\begingroup\n");
  printf("\\setlength\\LTleft{-90pt}\n");
  printf("\\setlength\\LTright{100pt}\n");
  printf("\\begin{longtable}{|p{0.2\\linewidth}|
  ↪  p{0.5\\linewidth}| p{0.25\\linewidth}| p{0.2\\linewidth}|
  ↪  p{0.15\\linewidth}|}\\hline\n");
```

```
  printf("\\textbf{Palavra} & \\textbf{Significado} &
  ↪  \\textbf{Variações}");
  printf(" & \\textbf{Designação em Inglês} &
  ↪  \\textbf{Sinónimos}\\\\\\hline\n");
  for(i=0; i<tamArray; i++){
          printInfo(dicionario,existentes[i]);
  }
  printf("\\caption{Tabela ilustrativa do Dicionário}\n");
  printf("\\end{longtable}\n");
  printf("\\endgroup\n");
  printf("\\end{appendix}\n");
}
  printf("\\end{document}");
```

Para simplificar o programa, evitando código repetido e, deste modo, mais limpo, foram criadas algumas funções auxiliares.

- **pontuacaoDif**, verifica se uma certa posição do *yytext* não tem um caracter de pontuação;
- **pontuacaoIgual**, verifica se uma certa posição do *yytext* tem um caracter de pontuação;
- **existef**, averigua se já existe uma determinada entrada no apêndice;
- **naoexiste**, adiciona uma entrada no apêndice;
- **iniciar**, coloca na variável *atual* a próxima palavra a comparar e no *pontAtual* os caracteres de pontuação a seguir a essa palavra.

3.4 Sistema de Tratamento de Textos Cinematográficos

Tendo a estrutura e todos os ficheiros *flex* e *yacc* desenvolvidos, foi necessário criar um programa capaz de juntar estas diferentes componentes, surgindo assim o STTC.

O STTC é um programa simples, onde apenas se encorporou a *main* que evoca os diferentes ficheiros. Esta começa por verificar que o número de argumentos passado é correto, pois é necessário aceder ao nome do ficheiro onde se encontra o dicionário cinematográfico e ao nome do ficheiro com o texto a processar. De seguida, cria-se uma estrutura *dicionario* vazia, com 40 posições. Esta estrutura é enviada na função *load* do *dicionario.y*, onde será realizado o *yyparse()* e preenchida a estrutura com os dados do dicionário. Por último, a estrutura já carregada é passada como parâmetro na função *lexTexto* do *texto.l*, onde é realizado o *yylex()* e processado o texto pretendido.

```
int main(int argc, char** argv){
    if(argc < 3){
        printf("Número de argumentos inválido.\n");
        return -1;
    }

    Dic dicionario;
    dicionario = init(40);
    dicionario = load(dicionario, argv[1]);
        lexTexto(dicionario, argv[2]);

    return 0;
}
```

3.5 Extras

Neste projeto foram desenvolvidos dois extras, sendo estes as variações de cada palavra do dicionário e o processamento de *verbatim*.

As variações são um extra que o grupo considerou essencial. Isto porque, tendo em conta que nos textos a mesma palavra pode aparecer no plural ou singular, feminino ou masculino, em verbo ou na forma comum, entre outras, esta foi a maneira que se considerou mais eficiente de lidar com todas as variações da palavra, evitando assim informação repetida no dicionário.

Relativamente ao *verbatim*, tendo em consideração que em *LaTeX* o *verbatim* é utilizado para evitar o processamento do texto contido dentro deste, o grupo considerou que esta caraterística devia ser mantida. Assim, o texto que aparece dentro de *verbatim* apenas é imprimido, sem qualquer verificação de palavras. Este foi um extra que se considerou fundamental para manter a consistência dos ficheiros *tex*.

4 Exemplos de Utilização

4.1 Latex com Verbatim

O seguinte código apenas contém Verbatim, logo não há rodapé nem apêndice.

```
\documentclass{article}
\usepackage[utf8]{inputenc}
\usepackage[portuges]{babel}
\begin{document}

\begin{verbatim}
isto não deve ser processado como um ator.
\end{verbatim}

\end{document}
```

O resultado após o processamento é:

```
\documentclass{article}
\usepackage[utf8]{inputenc}
\usepackage[portuges]{babel}
\usepackage[toc,page]{appendix}
\usepackage{array}
\newcolumntype{P}[1]{>{\hspace{0pt}}p{#1}}
\usepackage{longtable}
\begin{document}

\begin{verbatim}
isto não deve ser processado como um ator.
\end{verbatim}

\end{document}
```

4.2 Latex com uma e duas palavras

O seguinte código contém entradas do dicionário com uma e duas palavras. Além disso, tal como se pode verificar, algumas palavras apresentam-se, de forma propositada, com um uma mistura de maiúsculas e minúsculas para assim se poder verificar que estas são reconhecidas de igual forma como palavras cinematográficas.

```
\documentclass{article}
\usepackage[utf8]{inputenc}
\usepackage[portuges]{babel}
\begin{document}

Olá, eu sou um ator de primeira. Já disse, um \emph{ATOR}. Não é
↪  atOr secUNdáRiO.
Vamos fazer uma curta-metragem?
A banda sonora tem de ser boa e os efeitos visuais também.

\end{document}
```

O resultado após o processamento é:

```
\documentclass{article}
\usepackage[utf8]{inputenc}
\usepackage[portuges]{babel}
\usepackage[toc,page]{appendix}
\usepackage{array}
\newcolumntype{P}[1]{>{\hspace{0pt}}p{#1}}
\usepackage{longtable}
\begin{document}
```

```
Olá, eu sou um \textbf{ator}\footnote{actor} de primeira. Já
↪  disse, um \emph{\textbf{ATOR}\footnote{actor}}. Não é
↪  \textbf{atOr secUNdáRiO}\footnote{secondary actor}.
Vamos fazer uma \textbf{curta-metragem}\footnote{short film}?
A \textbf{banda sonora}\footnote{soundtrack} tem de ser boa e os
↪  \textbf{efeitos visuais}\footnote{visual effect} também.

\clearpage
\begin{appendix}
\section{Dicionário Cinematográfico}
\begingroup
\setlength\LTleft{-90pt}
\setlength\LTright{100pt}
\begin{longtable}{|p{0.2\linewidth}| p{0.5\linewidth}|
↪  p{0.25\linewidth}| p{0.2\linewidth}|
↪  p{0.15\linewidth}|}\hline
\textbf{Palavra} & \textbf{Significado} & \textbf{Variações} &
↪  \textbf{Designação em Inglês} & \textbf{Sinónimos}\\\hline
ator & Pessoa que interpreta um papel, encarnando uma personagem.
↪  & atriz; atores; atrizes & actor & -\\\hline
ator secundário & Pessoa que atua num papel secundário,
↪  contracenando direta ou indiretamente com os protagonistas. &
↪  atriz secundária; atores secundários; atrizes secundárias &
↪  secondary actor & -\\\hline
curta-metragem & Filme de duração geralmente inferior a trinta
↪  minutos.  & - & short film & -\\\hline
banda sonora & Gravação musical utilizada num filme. & - &
↪  soundtrack & -\\\hline
efeito visual & Nome dado a qualquer uma de várias técnicas
↪  utilizadas na indústria de cinema para realizar cenas que não
↪  podem ser obtidas por meios normais ou por ação ao vivo.  &
↪  efeitos visuais & visual effect & efeito especial\\\hline
\caption{Tabela ilustrativa do Dicionário}
\end{longtable}
\endgroup
\end{appendix}
\end{document}
```

4.3 Latex com Variações

O seguinte código contém variações de palavras que devem ser detetadas e, após isso, imprimir a entrada da tabela correspondente.

```
\documentclass{article}
\usepackage[utf8]{inputenc}
\usepackage[portuges]{babel}
\begin{document}

A atriz secundária fez um péssimo trabalho. Precisa de enquadrar
↪  melhor nas cenas.
Os realizadores não gostaram das representações dela no geral.

\end{document}
```

O resultado após o processamento é:

```
\documentclass{article}
\usepackage[utf8]{inputenc}
\usepackage[portuges]{babel}
\usepackage[toc,page]{appendix}
\usepackage{array}
\newcolumntype{P}[1]{>{\hspace{0pt}}p{#1}}
\usepackage{longtable}
\begin{document}

Olá, eu sou um \textbf{ator}\footnote{actor} de primeira. Já
↪  disse, um \emph{\textbf{ATOR}\footnote{actor}}. Não é
↪  \textbf{atOr secUNdáRiO}\footnote{secondary actor}.
Vamos fazer uma \textbf{curta-metragem}\footnote{short film}?
A \textbf{banda sonora}\footnote{soundtrack} tem de ser boa e os
↪  \textbf{efeitos visuais}\footnote{visual effect} também.

\clearpage
\begin{appendix}
\section{Dicionário Cinematográfico}
\begingroup
\setlength\LTleft{-90pt}
\setlength\LTright{100pt}
\begin{longtable}{|p{0.2\linewidth}| p{0.5\linewidth}|
↪  p{0.25\linewidth}| p{0.2\linewidth}|
↪  p{0.15\linewidth}|}\hline
\textbf{Palavra} & \textbf{Significado} & \textbf{Variações} &
↪  \textbf{Designação em Inglês} & \textbf{Sinónimos}\\\hline
ator & Pessoa que interpreta um papel, encarnando uma personagem.
↪  & atriz; atores; atrizes & actor & -\\\hline
```

```
ator secundário & Pessoa que atua num papel secundário,
↪  contracenando direta ou indiretamente com os protagonistas. &
↪  atriz secundária; atores secundários; atrizes secundárias &
↪  secondary actor & -\\\hline
curta-metragem & Filme de duração geralmente inferior a trinta
↪  minutos.  & - & short film & -\\\hline
banda sonora & Gravação musical utilizada num filme. & - &
↪  soundtrack & -\\\hline
efeito visual & Nome dado a qualquer uma de várias técnicas
↪  utilizadas na indústria de cinema para realizar cenas que não
↪  podem ser obtidas por meios normais ou por ação ao vivo.  &
↪  efeitos visuais & visual effect & efeito especial\\\hline
\caption{Tabela ilustrativa do Dicionário}
\end{longtable}
\endgroup
\end{appendix}
\end{document}
```

4.4 Latex Geral

O exemplo apresentado de seguida contém os vários casos expostos acima.

```
\documentclass{article}
\usepackage[utf8]{inputenc}
\usepackage[portuges]{babel}
\begin{document}

Olá, eu sou um ator de primeira. Já disse, um \emph{ATOR}. Não é
↪  atOr secUNdáRiO.
Eu exerço a arte da Representação.
Vamos fazer uma curta-metragem?
\begin{verbatim}
isto não deve ser processado como um ator.
\end{verbatim}
Nem preciso de \textbf{duplo}, atuar é comigo em todos os
↪  cenários.
O nosso filme tem os melhores efeitos visuais do ano.

\end{document}
```

O resultado após o processamento é:

```
\documentclass{article}
\usepackage[utf8]{inputenc}
\usepackage[portuges]{babel}
\usepackage[toc,page]{appendix}
\usepackage{array}
```

```
\newcolumntype{P}[1]{>{\hspace{0pt}}p{#1}}
\usepackage{longtable}
\begin{document}

Olá, eu sou um \textbf{ator}\footnote{actor} de primeira. Já
↪ disse, um \emph{\textbf{ATOR}\footnote{actor}}. Não é
↪ \textbf{atOr secUNdáRiO}\footnote{secondary actor}.
Eu exerço a arte da \textbf{Representação}\footnote{acting}.
Vamos fazer uma \textbf{curta-metragem}\footnote{short film}?
\begin{verbatim}
isto não deve ser processado como um ator.
\end{verbatim}
Nem preciso de \textbf{\textbf{duplo}\footnote{double}},
↪ \textbf{atuar}\footnote{acting} é comigo em todos os
↪ \textbf{cenários}\footnote{scenario}.
O nosso \textbf{filme}\footnote{movie} tem os melhores
↪ \textbf{efeitos visuais}\footnote{visual effect} do ano.

\clearpage
\begin{appendix}
\section{Dicionário Cinematográfico}
\begingroup
\setlength\LTleft{-90pt}
\setlength\LTright{100pt}
\begin{longtable}{|p{0.2\linewidth}| p{0.5\linewidth}|
↪ p{0.25\linewidth}| p{0.2\linewidth}|
↪ p{0.15\linewidth}|}\hline
\textbf{Palavra} & \textbf{Significado} & \textbf{Variações} &
↪ \textbf{Designação em Inglês} & \textbf{Sinónimos}\\\hline
ator & Pessoa que interpreta um papel, encarnando uma personagem.
↪ & atriz; atores; atrizes & actor & -\\\hline
ator secundário & Pessoa que atua num papel secundário,
↪ contracenando direta ou indiretamente com os protagonistas. &
↪ atriz secundária; atores secundários; atrizes secundárias &
↪ secondary actor & -\\\hline
representação & Desempenhar um papel. & representar;
↪ representações; atuar & acting & atuação\\\hline
curta-metragem & Filme de duração geralmente inferior a trinta
↪ minutos. & - & short film & -\\\hline
duplo & Pessoa que substitui um ator ou uma atriz em determinadas
↪ cenas. & dupla; duplos; duplas & double & substituto\\\hline
cenário & Conjunto dos bastidores e vistas apropriadas aos factos
↪ que se representam; lugar onde decorre um acontecimento, uma
↪ ação. & cenários & scenario & plano de fundo; paisagem
↪ \\\hline
```

```
filme & Produto audiovisual finalizado, com uma certa duração,
 ↪  para ser exibido no cinema. & filmes & movie &
 ↪  película\\\hline
efeito visual & Nome dado a qualquer uma de várias técnicas
 ↪  utilizadas na indústria de cinema para realizar cenas que não
 ↪  podem ser obtidas por meios normais ou por ação ao vivo.  &
 ↪  efeitos visuais & visual effect & efeito especial\\\hline
\caption{Tabela ilustrativa do Dicionário}
\end{longtable}
\endgroup
\end{appendix}
\end{document}
```

5 Conclusão

Atendendo ao objetivo deste trabalho, o desenvolvimento de um programa capaz de processar um dicionário e um texto e transformar o texto segundo as informações do dicionário, foi necessário aprofundar os nossos conhecimentos em vários temas e atacar o problema por vários lados.

Por um lado, aprofundamos os nosso conhecimentos em reconhecedores léxicos e sintáticos, nomeadamente o *flex* e o *yacc*, para extrair e processar as informações contidas nos ficheiros. Por outro lado, foi necessário aplicar os conhecimentos adquiridos anteriormente em C, para desenvolver uma estrutura de dados de raíz, neste caso, uma tabela de *hash*. Para além disso, foi também essencial perceber como se deveria compilar e executar os diversos ficheiros originados, pois inicialmente deparamo-nos com o problema de compilar e executar dois ficheiros *flex* distintos.

No desenvolvimento do trabalho teve-se especial atenção na gramática criada, pois o funcionamento desta é essencial para o sucesso do programa, e à correta inserção das informações na estrutura, devido ao cuidado que se deve ter com apontadores em C. Foi também prestada a atenção a promenores como a não inserção de informação repetida no apêndice criado no texto cinematográfico e na procura de palavras pela sua variação e não só pela palavra presente no dicionário.

É importante referir que a estrutura de dados criada é dinâmica, o que significa que, se no futuro se quiser expandir o dicionário adicionando mais entradas, o tamanho da estrutura é aumentado para suportar o número de entradas desejadas.

No geral, consideramos que cumprimos o objetivo do trabalho e que os extras criados são importantes para aumentar a eficácia do programa. Como trabalho futuro, poderiamos pensar em mais extras, como por exemplo o processamento de mais linguagens para além do *LaTeX*.