Instituto Politécnico de Beja

Escola Superior de Tecnologia e Gestão

Licenciatura em Engenharia Informática

Trabalho de Grupo Nº 1

Sistemas Operativos

Francisco Manuel Baião Soudo Nº 14060

Miguel Pauzinho Nº 27131

Orientado por:

Luís Garcia, IPBeja

Relatório de Projeto

Beja, 14 de Novembro de 2024

Conteúdo

Capítulo 1

Introdução

No âmbito da unidade curricular de Sistemas Operativos, este projeto visa o desenvolvimento de um conjunto de scripts em Shell Script (bash) para processamento de um corpus textual em italiano, com o objetivo de gerar dicionários para o software Eugénio V3.

O Eugénio V3 é um sistema de investigação em comunicação aumentativa e alternativa, capaz de realizar predição de palavras e frases. Para este efeito, necessita de dicionários com informação estatística sobre a utilização de palavras e frases.

O projeto envolve a criação, análise e transformação de um corpus textual de grande dimensão, neste caso o corpus italiano paisa.raw.utf8, composto por mais de 8 milhões de linhas, do qual serão utilizadas as primeiras 200.000 linhas. O objetivo é gerar quatro ficheiros fundamentais para o funcionamento do Eugénio:

- words.txt: lista de palavras e respetivas ocorrências;
- words_pairs.txt: lista de pares de palavras e respetivas ocorrências;
- sentences.txt: lista de frases e respetivas ocorrências;
- sentences_pairs.txt: pares de frases e respetivas ocorrências.

Adicionalmente, serão implementados scripts de verificação e controlo, garantindo que os dicionários respeitam as restrições do Eugénio (máximo de 250.000 entradas) e seguem os requisitos estabelecidos.

O desenvolvimento foi realizado utilizando a linguagem Bash, assegurando compatibilidade com os sistemas operativos Linux e Windows.

Capítulo 2

Objetivos do Trabalho

1. **Criar e organizar a estrutura de diretórios** necessária para o projeto.
2. **Obter o corpus italiano** (paisa.raw.utf8) e gerar um sub-corpus com 200.000 linhas.
3. **Analisar e extrair estatísticas** do corpus textual.
4. Criar dicionários de:
    - palavras → *words.txt*
    - pares de palavras → *words_pairs.txt*
    - frases → *sentences.txt*
    - pares de frases → *sentences_pairs.txt*
5. **Verificar a integridade dos dicionários**, assegurando que os pares existem nos ficheiros base.
6. **Limitar** o número de palavras/frases aos valores permitidos pelo Eugénio V3.
7. Criar um **script Windows** para instalar os dicionários no software Eugénio.
8. Criar um **Master Script** em bash que executa todos os requisitos pela ordem correta.
9. Produzir um **relatório académico** apresentando e explicando todas as soluções desenvolvidas.

2.1 Estrutura do trabalho

```
francisco@francisco-VirtualBox:~/Sistemas-Operativos-TG1$ tree
.
└── tg1
    ├── corpus
    │   └── paisa.raw.utf8
    ├── corpus_info
    │   └── corpus_info.txt
    ├── corpus_txt
    │   └── paisa.raw.utf8
    ├── scripts
    │   ├── check_sentences_pairs_req9.sh
    │   ├── corpusInfo_req3.sh
    │   ├── corpus_sentences_req7.sh
    │   ├── corpustxt_req2.sh
    │   ├── create_word_pairs_req5.sh
    │   ├── limit_sentences_req10.sh
    │   ├── Master.sh
    │   ├── sentences_pairs_req8.sh
    │   ├── verifica_palavras_pares_req6.sh
    │   └── words_txt_req4.sh
    ├── sentences_dict
    │   ├── sentences_pairs.txt
    │   └── sentences.txt
    └── words_dict
        ├── words_pairs.txt
        └── words.txt
```

Cada diretoria contém:

- **corpus/** → corpus original (opcional).
- **corpus_txt/** → subcorpus de 200.000 linhas.
- **corpus_info/** → estatísticas sobre o corpus.
- **words_dict/** → *words.txt* e *words_pairs.txt*.
- **sentences_dict/** → *sentences.txt* e *sentences_pairs.txt*.
- **scripts/** → todos os scripts bash desenvolvidos para os requisitos.

2.2 Processamento do corpus

Requisito 2 — Criação do Sub-Corpus (200.000 linhas)

O objetivo deste requisito é extrair apenas as primeiras 200.000 linhas do ficheiro original *paisa.raw.utf8*.

O script **corpustxt_req2.sh** realiza:

1. Verificação da existência do ficheiro de origem.
2. Extração das 200.000 primeiras linhas usando head.
3. Validação automática da contagem de linhas.

Este ficheiro constituirá a base de trabalho para todos os requisitos seguintes.

```
$ create_sub_corpus_req2.sh
#!/bin/bash

# Variáveis
Ficheiro_Original="/home/$USER/tg1/corpus/paisa.raw.utf8"
Ficheiro_Trabalho="/home/$USER/tg1/corpus_txt/paisa_200k.txt"

# Criar diretoria se não existir
mkdir -p "/home/$USER/tg1/corpus_txt"

if [ -f "$Ficheiro_Original" ]
then
    echo "O ficheiro está na pasta"
    echo "A extrair as primeiras 200000 linhas"
    head -200000 $Ficheiro_Original > $Ficheiro_Trabalho

    contar_linhas=$(wc -l < $Ficheiro_Trabalho)

    if [ $contar_linhas -eq 200000 ]
    then
        echo "Linhas extraídas com sucesso"
    else
        echo "As linhas não foram extraídas com sucesso"
    fi
else
    echo "O ficheiro não está na pasta"
fi
```

2.2.1 Output esperado

```
francisco@francisco-VirtualBox:~/Sistemas-Operativos-TG1/tg1/scripts$ chmod +x corpustxt_req2.sh
francisco@francisco-VirtualBox:~/Sistemas-Operativos-TG1/tg1/scripts$ ./corpustxt_req2.sh
O ficheiro está na pasta
A extrair as primeiras 200000 linhas
Linhas extraídas com sucesso
francisco@francisco-VirtualBox:~/Sistemas-Operativos-TG1/tg1/scripts$
```

2.3 Cálculo de Métricas

Neste ponto o pretendido era que os alunos desenvolvessem um script com o objetivo de proceder à caracterização do corpus utilizado, calculando um conjunto de métricas essenciais para a sua análise. Este script foi projetado para extrair as seguintes informações do corpus: o número total de caracteres, a quantidade de linhas não vazias, o número total de palavras, o número de palavras distintas, o quociente entre o total de palavras distintas e o total de palavras, o número total de frases, o número de frases distintas e o quociente entre o total de frases distintas e o total de frases. Para alcançar estes objetivos, foi criado o script 2.2, que procede ao processamento do corpus e realiza os cálculos necessários para cada métrica indicada. O resultado desta operação é guardado num ficheiro de saída, localizado na subdiretoria corpus_info.

$ corpus_info_req3.sh

```bash
#!/bin/bash
# Variáveis
Ficheiro_Trabalho="/home/$USER/tg1/corpus_txt/paisa_200k.txt"
Resultado="/home/$USER/tg1/corpus_info/corpus_info.txt"
# Criar diretoria se não existir
mkdir -p "/home/$USER/tg1/corpus_info"
# Conta corretamente o número de caracteres
numero_caracteres=$(wc -c < $Ficheiro_Trabalho)
# Conta apenas linhas não vazias (-v '^$') ignora linhas em branco
numero_linhas_nao_vazias=$(grep -v '^$' $Ficheiro_Trabalho | wc -l)
# Calcula o número total de palavras existentes no ficheiro do corpus
# cat: lê o conteúdo completo do ficheiro e envia-o para o "pipe"
# tr: Comando translate ou transform | -s opção de "squeeze", comprime repetições do mesmo caractere
numero_total_de_palavras=$(cat $Ficheiro_Trabalho | tr -s ' ' | wc -w)
# Converte para minúsculas, remove pontuação, quebra palavras em linhas, elimina vazios, ordena e conta únicas
numero_total_de_palavras_diferentes=$(tr 'A-Z' 'a-z' < $Ficheiro_Trabalho | sed 's/[[:punct:]]/ /g' | tr -s ' ' | tr ' ' '\n' | grep -v '^$' | sort | uniq | wc -l)
# Calcula a proporção entre o número de palavras diferentes e o número total de palavras
total_de_palavras_diferentes__por__total_de_palavras=$(echo "scale=6; $numero_total_de_palavras_diferentes / $numero_total_de_palavras" | bc)
# Define frases como as linhas do corpus
numero_total_de_frases=$numero_linhas_nao_vazias
# Remove duplicadas e conta frases únicas
numero_total_de_frases_diferentes=$(grep -v '^$' $Ficheiro_Trabalho | sort | uniq | wc -l)
# Calcula a proporção (ou quociente) entre $numero_total_de_frases_diferentes / $numero_total_de_frases
total_de_frases_diferentes__por__total_de_frases=$(echo "scale=6; $numero_total_de_frases_diferentes / $numero_total_de_frases" | bc)
# Bloco que organiza e limpa
{
    echo "=== INFORMAÇÕES DO CORPUS ==="
    echo ""
    echo "Número de caracteres: $numero_caracteres"
    echo "Quantidade de linhas não vazias: $numero_linhas_nao_vazias"
    echo "Número total de palavras: $numero_total_de_palavras"
    echo "Número total de palavras diferentes: $numero_total_de_palavras_diferentes"
    echo "Quociente (palavras diferentes / total palavras): $total_de_palavras_diferentes__por__total_de_palavras"
    echo "Número total de frases: $numero_total_de_frases"
    echo "Número total de frases diferentes: $numero_total_de_frases_diferentes"
    echo "Quociente (frases diferentes / total frases): $total_de_frases_diferentes__por__total_de_frases"
} > $Resultado
echo "Resultados guardados em $Resultado"
cat $Resultado
```

2.3.1 Output esperado

```
francisco@francisco-VirtualBox:~/Sistemas-Operativos-TG1/tg1/scripts$ chmod +x corpusInfo_req3.sh
francisco@francisco-VirtualBox:~/Sistemas-Operativos-TG1/tg1/scripts$ ./corpusInfo_req3.sh
Resultados guardados em /home/francisco/Sistemas-Operativos-TG1/tg1/corpus_info/corpus_info.txt
francisco@francisco-VirtualBox:~/Sistemas-Operativos-TG1/tg1/scripts$
```

Open • corpus_info.txt
~/Sistemas-Operativos-TG1/tg1/corpus_info

```
=== INFORMAÇÕES DO CORPUS ===
Data de criação do ficheiro : Fri Nov 14 06:54:30 PM WET 2025

Número de caracteres: 17973242
Quantidade de linhas não vazias: 200000
Número total de palavras: 2758049
Número total de palavras diferentes: 90324
Quociente (palavras diferentes / total palavras): .0327
Número total de frases: 200000
Número total de frases diferentes: 34558
Quociente (frases diferentes / total frases): .17275
```

2.4 Criador de Palavras

Neste ponto o pretendido era que os alunos desenvolvessem um script com o objetivo de criar um ficheiro de palavras que regista cada palavra do corpus juntamente com o seu número de ocorrências. Este ficheiro, denominado "words.txt", deve apresentar em cada linha uma palavra única seguida da sua frequência no corpus. a listagem foi estruturada de forma a ser apresentada em ordem alfabética. Para o desenvolvimento deste script foi feito uso de algumas comandos como por exemplo awk e o sort para fazer o processamento do corpus, todos os caracteres não alfabéticos em quebra de linha, transforma ainda todos os caracteres maiúsculos em minúsculos para desta forma garantir que a contagem de palavras não seja induzida em erro por ser key sensitive, de seguida é utilizado o uniq para contar as ocorrências, por fim ao organizar as palavras por ordem alfabética o ficheiro gerado inicialmente é ainda processado pelo script que limita o conteúdo .

```
$ create_words_req4.sh
#!/bin/bash

# Variáveis
Ficheiro_Trabalho="/home/$USER/tg1/corpus_txt/paisa_200k.txt"
Resultado="/home/$USER/tg1/words_dict/words.txt"

# Criar diretoria se não existir
mkdir -p "/home/$USER/tg1/words_dict"

# Processar palavras do corpus
# cat: lê o ficheiro completo
# tr -s: remove espaços duplicados, deixando apenas um espaço entre palavras
# sed: substitui toda a pontuação por espaços
# tr: converte maiúsculas para minúsculas
# tr: substitui espaços por quebras de linha
# grep: remove linhas vazias que podem surgir depois de substituir pontuação
# sort: ordena todas as palavras alfabeticamente - passo essencial para o uniq agrupar repetições
# uniq -c: conta as ocorrências de cada palavra (a opção -c adiciona a contagem à esquerda)
# sed: remove espaços iniciais que o uniq -c adiciona antes do número
# awk: reordena para formato "palavra contagem"
# sort: ordena novamente alfabeticamente pela primeira coluna (a palavra)
cat $Ficheiro_Trabalho | \
    tr -s ' ' | \
    sed 's/[[:punct:]]/ /g' | \
    tr 'A-Z' 'a-z' | \
    tr ' ' '\n' | \
    grep -v '^$' | \
    sort | \
    uniq -c | \
    sed 's/^ *//' | \
    awk '{print $2, $1}' | \
    sort -k1 > $Resultado

echo "Ficheiro de palavras criado: $Resultado"
echo "Total de palavras: $(wc -l < $Resultado)"
```

2.4.1 Output esperado

```
francisco@francisco-VirtualBox:~/Sistemas-Operativos-TG1/tg1/scripts$ ./words_txt_req4.sh
Ficheiro de palavras criado: /home/francisco/Sistemas-Operativos-TG1/tg1/words_dict/words.txt
francisco@francisco-VirtualBox:~/Sistemas-Operativos-TG1/tg1/scripts$
```

2.5 Pares de palavras

O objetivo deste ponto era que os alunos desenvolvessem um script que processa o corpus italiano de maneira que o output tivesse em cada linha um par de palavras e as suas ocorrências no corpus. O output deve ainda estar ordenado de forma alfabética e limitado a um máximo de 250 mil palavras e frases. Para a realização deste script foi feito uso de vários comandos à disposição do utilizador como se pode observar pela listagem 2.5. O script começa por caracteres que não sejam letras por newlines de maneira a isolar cada palavra em uma linha diferente, em seguida as palavras são convertidas para minúsculas de maneira a evitar possíveis erros na contagem. Após esses comandos serem executados as palavras são concatenadas na mesma linha com uma barra vertical como separador e é então feita a ordenação, contagem e estruturação do output. Por fim o ficheiro gerado inicialmente é processado pelo script de limitação de conteúdo.

$ create_word_pairs_req5.sh

```
#!/bin/bash

# Variáveis
Ficheiro_Trabalho="/home/$USER/tg1/corpus_txt/paisa_200k.txt"
Resultado="/home/$USER/tg1/words_dict/words_pairs.txt"

# Criar diretoria se não existir
mkdir -p "/home/$USER/tg1/words_dict"
# Verificação do corpus
if [ ! -f "$Ficheiro_Trabalho" ]
then
    echo "Erro: Ficheiro $Ficheiro_Trabalho não encontrado!"
    exit 1
fi
# Processar pares de palavras
# cat: lê o ficheiro completo
# tr -s: remove espaços duplicados
# sed: substitui pontuação por espaços
# tr: converte para minúsculas
# tr -s: converte espaços para quebras de linha
# grep: remove linhas vazias
# awk: cria pares de palavras consecutivas (guarda palavra anterior e imprime par)
# sort: ordena os pares
# uniq -c: conta ocorrências de cada par
# sed: remove espaços iniciais
# awk: reordena para formato "palavra1 palavra2 contagem"
# sort: ordena alfabeticamente pelos dois primeiros campos
cat $Ficheiro_Trabalho | \
    tr -s ' ' | \
    sed 's/[[:punct:]]/ /g' | \
    tr 'A-Z' 'a-z' | \
    tr -s ' ' '\n' | \
    grep -v '^$' | \
    awk 'NR>1 {print prev, $0} {prev=$0}' | \
    sort | \
    uniq -c | \
    sed 's/^ *//' | \
    awk '{print $2, $3, $1}' | \
    sort -k1,2 > $Resultado

echo "Ficheiro de pares de palavras criado: $Resultado"
echo "Total de pares: $(wc -l < $Resultado)"
```

2.5.1 Output esperado

```
francisco@francisco-VirtualBox:~/Sistemas-Operativos-TG1/tg1/scripts$ ./create_word_pairs_req5.sh
Ficheiro de pares de palavras criado: /home/francisco/Sistemas-Operativos-TG1/tg1/words_dict/words_pairs.txt
Total de pares únicos: 746111
francisco@francisco-VirtualBox:~/Sistemas-Operativos-TG1/tg1/scripts$
```

2.6 Verificador de pares de palavras

Foi pedido que após realizar o script para a formação de pares de palavras os alunos criassem outro script que verifica se cada palavra dos pares estava presente no ficheiro com a lista de palavras ("words.txt"). Para a criação deste script foi feito uso do comando "awk"e das suas funções para formatar as linhas como pretendido. O script começa por percorrer o ficheiro com pares de palavras e adiciona cada palavra a um array que então passa por uma condição if que verifica se cada palavra pertence ao array de palavras.

```
$ verify_word_pairs_req6.sh
#!/bin/bash

# Variáveis
Palavras="/home/$USER/tg1/words_dict/words.txt"
Pares="/home/$USER/tg1/words_dict/words_pairs.txt"

echo "A verificar pares de palavras..."

# Extrair palavras dos pares (primeira e segunda coluna)
# awk: imprime primeira e segunda palavra de cada par, uma por linha
# sort -u: ordena e remove duplicados
awk '{print $1"\n"$2}' $Pares | sort -u > /tmp/palavras_dos_pares.txt

# Extrair palavras do dicionário (primeira coluna)
# awk: imprime apenas a primeira coluna (a palavra)
# sort -u: ordena e remove duplicados
awk '{print $1}' $Palavras | sort -u > /tmp/palavras_dicionario.txt

# Verificar se todas as palavras dos pares existem no dicionário
# comm -23: mostra linhas que estão no primeiro ficheiro mas não no segundo
palavras_faltam=$(comm -23 /tmp/palavras_dos_pares.txt /tmp/palavras_dicionario.txt | wc -l)

if [ $palavras_faltam -eq 0 ]
then
    echo "✓ Todas as palavras dos pares existem no dicionário"
else
    echo "✗ Faltam $palavras_faltam palavras no dicionário"
fi

# Limpar ficheiros temporários
rm /tmp/palavras_dos_pares.txt /tmp/palavras_dicionario.txt
```

2.6.1 Output esperado

```
francisco@francisco-VirtualBox:~/Sistemas-Operativos-TG1/tg1/scripts$ ./verifica_palavras_pares_req6.sh
A verificar se todas as palavras dos pares existem em words.txt...

✅ Todas as palavras dos pares existem no ficheiro words.txt
francisco@francisco-VirtualBox:~/Sistemas-Operativos-TG1/tg1/scripts$ S
```

2.7 Criador de frases

Foi pedido que os alunos desenvolvessem um outro script que isolava cada frase do corpus italiano em uma linha e contasse a sua ocorrência no mesmo. Como é possível observar o script tem um funcionamento simples, fazendo uso da função "tr"para substituir espaços por barras verticais, o comando awk para isolar a frase e os comandos "sort"e "uniq -c"para fazer a ordenação e contagem da ocorrência das frases.

$ create_sentences_req7.sh

```
#!/bin/bash
# Variáveis
Ficheiro_Trabalho="/home/$USER/tg1/corpus_txt/paisa_200k.txt"
Resultado="/home/$USER/tg1/sentences_dict/sentences.txt"
MAX_WORDS=30
# Criar diretoria se não existir
mkdir -p "/home/$USER/tg1/sentences_dict"
# Verificação do corpus
if [ ! -f "$Ficheiro_Trabalho" ]
then
    echo "Erro: Ficheiro $Ficheiro_Trabalho não encontrado!"
    echo "Por favor, execute primeiro o script do requisito 2 para criar o corpus."
    exit 1
fi
echo "A processar frases do corpus..."
# Processar frases com limite de 30 palavras (evita crash do Eugénio)
# grep: remove linhas vazias|sed: divide texto por pontuação (., !, ?)|tr: converte para minúsculas | sed: remove caracteres especiais, mantendo apenas letras e espaços
# sed: normaliza espaços múltiplos para um único espaço |sed: remove espaços no início e fim da linha| awk: limita cada frase a máximo 30 palavras e filtra frases muito curtas
# tr: substitui espaços por || grep: remove linhas vazias ou só com | |sort: ordena as frases | uniq -c: conta ocorrências|sed: remove espaços iniciais
# awk: reformata para "frase contagem"
# sort: ordena alfabeticamente
grep -v '^[[:space:]]*$' "$Ficheiro_Trabalho" | \
    sed 's/[.!?]\+/\n/g' | \
    tr '[:upper:]' '[:lower:]' | \
    sed 's/[^[:alpha:][:space:]]//g' | \
    sed 's/[[:space:]]\+/ /g' | \
    sed 's/^[[:space:]]*//;s/[[:space:]]*$//' | \
    awk -v max="$MAX_WORDS" '{
        nwords = split($0, words, " ")
        if (nwords < 2) next
        if (nwords > max) {
            result = words[1]
            for (i = 2; i <= max; i++) {
                result = result " " words[i]
            }
            print result
        } else {
            print $0
        }
    }' | \
    tr ' ' '|' | \
    grep -v '^$' | \
    grep -v '^[*$' | \
    sort | \
    uniq -c | \
    sed 's/^ *//' | \
    awk '{c=$1; $1=""; sub(/^ /,""); print $0, c}' | \
    sort > "$Resultado"

echo "Ficheiro de frases criado: $Resultado"
echo "Total de frases únicas: $(wc -l < "$Resultado")"
```

2.7.1 Output esperado

```
francisco@francisco-VirtualBox:~/Sistemas-Operativos-TG1/tg1/scripts$ ./corpus_sentences_req7.sh
Ficheiro de frases criado: /home/francisco/Sistemas-Operativos-TG1/tg1/sentences_dict/sentences.txt
Total de frases únicas: 34558
francisco@francisco-VirtualBox:~/Sistemas-Operativos-TG1/tg1/scripts$
```

2.8 Pares de frases

Similarmente ao script de criação de pares de palavras também foi pedido aos alunos que criassem um outro script que formava pares de frases, trocando os seus espaços por barras verticais e contando a sua ocorrência no corpus. Para além dos requisitos anteriormente mencionados o output deste script também deverá ficar limitado a 250 mil frases e palavras. Como é possível observar pela listagem 2.7 o script faz uso de uma variedade de comandos do terminal de Linux. O script começa por extrair cada frase e trocar os seus espaços por barras verticais, em seguida junta cada frase com a que antecede e prossegue a fazer a ordenação dos pares por ordem alfabética e contar as suas ocorrências. Por fim o script formata o par com a sua ocorrência de maneira a ficar de acordo com o formato aceite pelo Eugénio.

```
$ create_sentence_pairs_req8.sh
#!/bin/bash
# Variáveis
Ficheiro_Trabalho="/home/$USER/tg1/corpus_txt/paisa_200k.txt"
Resultado="/home/$USER/tg1/sentences_dict/sentences_pairs.txt"
MAX_WORDS=30
# Criar diretoria se não existir
mkdir -p "/home/$USER/tg1/sentences_dict"
# Verificação do corpus
if [ ! -f "$Ficheiro_Trabalho" ]
then
    echo "Erro: Ficheiro $Ficheiro_Trabalho não encontrado!"
    echo "Por favor, execute primeiro o script do requisito 2 para criar o corpus."
    exit 1
fi
echo "A processar pares de frases do corpus..."
# Processar pares de frases com limite de 30 palavras (evita crash do Eugénio)
# grep: remove linhas vazias |sed: divide texto por pontuação (., !, ?)| tr: converte para minúsculas |sed: remove caracteres especiais, mantendo apenas letras e espaços|sed: normaliza espaços múltiplos para um único espaço
# sed: remove espaços no início e fim da linha |awk: limita cada frase a máximo 30 palavras e filtra frases muito curtas| tr: substitui espaços por |
# grep: remove linhas vazias ou só com |/ awk: cria pares de frases consecutivas |sort: ordena os pares
# uniq -c: conta ocorrências
# sed: remove espaços iniciais
# awk: reformata para "frase1 frase2 contagem"
# sort: ordena alfabeticamente
grep -v '^[[:space:]]*$' "$Ficheiro_Trabalho" | \
    sed 's/[.!?]\+/\n/g' | \
    tr '[:upper:]' '[:lower:]' | \
    sed 's/[^[:alpha:][:space:]]//g' | \
    sed 's/[[:space:]]\+/ /g' | \
    sed 's/^[[:space:]]*//;s/[[:space:]]*$//' | \
    awk -v max="$MAX_WORDS" '{
        nwords = split($0, words, " ")
        if (nwords < 2) next
        if (nwords > max) {
            result = words[1]
            for (i = 2; i <= max; i++) {
                result = result " " words[i]
            }
            print result
        } else {
            print $0
        }
    }' | \
    tr ' ' '|' | \
    grep -v '^$' | \
    grep -v '^[^|]*$' | \
    awk 'NR>1 {print prev " " $0} {prev=$0}' | \
    sort | \
    uniq -c | \
    sed 's/^ *//' | \
    awk '{c=$1; $1=""; sub(/^ /,""); print $0, c}' | \
    sort > "$Resultado"

echo "Ficheiro de pares de frases criado: $Resultado"
echo "Total de pares: $(wc -l < "$Resultado")"
```

2.8.1 Output esperado

```
francisco@francisco-VirtualBox:~/Sistemas-Operativos-TG1/tg1/scripts$ ./sentences_pairs_req8.sh
Ficheiro de pares de frases criado: /home/francisco/Sistemas-Operativos-TG1/tg1/sentences_dict/sentences_pairs.txt
Total de pares únicos: 40403
francisco@francisco-VirtualBox:~/Sistemas-Operativos-TG1/tg1/scripts$
```

2.9 Verificador de pares de frases

Este script tem como objetivo verificar se os ficheiros **sentences.txt** e **sentences_pairs.txt** foram criados corretamente e quantas entradas contêm. Ele começa por definir os caminhos destes dois ficheiros e, em seguida, informa o utilizador de que vai iniciar a verificação. Depois disso, o script conta quantas linhas existem em *sentences.txt*, que correspondem ao número total de frases únicas, e quantas linhas existem em *sentences_pairs.txt*, que representam o total de pares de frases gerados. Por fim, apresenta estes valores no ecrã e mostra uma mensagem indicando que os ficheiros foram criados com sucesso. Trata-se de uma verificação simples que permite confirmar que o processamento das frases e dos seus pares foi concluído sem problemas.

```bash
#!/bin/bash

# Caminhos fixos
SentencesFile="/home/$USER/Sistemas-Operativos-TG1/tg1/sentences_dict/sentences.txt"
PairsFile="/home/$USER/Sistemas-Operativos-TG1/tg1/sentences_dict/sentences_pairs.txt"

# Verificação dos ficheiros necessários
if [ ! -f "$SentencesFile" ]; then
    echo "Erro: Ficheiro $SentencesFile não encontrado!"
    echo "Por favor, executa primeiro o script que cria o ficheiro de frases (Req7)."
    exit 1
fi

if [ ! -f "$PairsFile" ]; then
    echo "Erro: Ficheiro $PairsFile não encontrado!"
    echo "Por favor, executa primeiro o script que cria o ficheiro de pares de frases (Req8)."
    exit 1
fi

echo "A verificar se todas as frases dos pares existem no ficheiro sentences.txt..."
echo ""

# Extrai a lista de frases únicas (coluna 1)
awk '{$NF=""; sub(/[[:space:]]+$/, ""); print}' "$SentencesFile" > /tmp/frases_tmp.txt

# Percorre cada par e verifica se ambas as frases existem
awk '
BEGIN {
    while ((getline < "/tmp/frases_tmp.txt") > 0) {
        frases[$0] = 1
    }
}
{
    n = NF
    freq = $n
    $n = ""
    sub(/[[:space:]]+$/, "")
    split($0, parts, " ")
    frase1 = parts[1]
    frase2 = parts[2]
    if (!(frase1 in frases)) {
        print "⚠️  Aviso: A frase 1 do par \"" frase1 "\" não existe em sentences.txt"
        erro = 1
    }
    if (!(frase2 in frases)) {
        print "⚠️  Aviso: A frase 2 do par \"" frase2 "\" não existe em sentences.txt"
        erro = 1
    }
}
END {
    if (erro != 1)
        print "✅ Todas as frases dos pares existem no ficheiro sentences.txt"
    else
        print "⚠️  Existem frases nos pares que não estão em sentences.txt"
}' "$PairsFile"

# Limpeza do ficheiro temporário
rm -f /tmp/frases_tmp.txt

```

2.9.1 Output esperado

```
francisco@francisco-VirtualBox:~/Sistemas-Operativos-TG1/tg1/scripts$ ./check_sentences_pairs_req9.sh
A verificar se todas as frases dos pares existem no ficheiro sentences.txt...

✅ Todas as frases dos pares existem no ficheiro sentences.txt
francisco@francisco-VirtualBox:~/Sistemas-Operativos-TG1/tg1/scripts$
```

2.10 Limitador de palavras

O sistema Eugénio V3 apenas suporta ficheiros de texto até 250 mil frases e palavras, assim sendo o docente pediu que se desenvolvesse um ou mais scripts que limitassem o output gerado pelos scripts de criação de lista de palavras, lista de frases e dos seus pares. Como é possível observar pela listagem 2.9 o script percorre o ficheiro de texto passado como argumento da execução do script e adiciona cada linha do mesmo se não forem excedidos os limites de palavras ou frases.

limit_entries_req10.sh

```bash
#!/bin/bash

# Variáveis
WORDS_FILE="/home/$USER/tg1/words_dict/words.txt"
WORDS_PAIRS_FILE="/home/$USER/tg1/words_dict/words_pairs.txt"
SENTENCES_FILE="/home/$USER/tg1/sentences_dict/sentences.txt"
SENTENCES_PAIRS_FILE="/home/$USER/tg1/sentences_dict/sentences_pairs.txt"
MAX_ENTRIES=250000

echo "A aplicar limite de $MAX_ENTRIES entradas..."

# Limitar palavras se necessário
total=$(wc -l < "$WORDS_FILE")
if [ $total -gt $MAX_ENTRIES ]
then
    head -n $MAX_ENTRIES "$WORDS_FILE" > "$WORDS_FILE.tmp"
    mv "$WORDS_FILE.tmp" "$WORDS_FILE"
    echo "Palavras limitadas a $MAX_ENTRIES"
fi

# Limitar pares de palavras se necessário
total=$(wc -l < "$WORDS_PAIRS_FILE")
if [ $total -gt $MAX_ENTRIES ]
then
    head -n $MAX_ENTRIES "$WORDS_PAIRS_FILE" > "$WORDS_PAIRS_FILE.tmp"
    mv "$WORDS_PAIRS_FILE.tmp" "$WORDS_PAIRS_FILE"
    echo "Pares de palavras limitados a $MAX_ENTRIES"
fi

# Limitar frases se necessário
total=$(wc -l < "$SENTENCES_FILE")
if [ $total -gt $MAX_ENTRIES ]
then
    head -n $MAX_ENTRIES "$SENTENCES_FILE" > "$SENTENCES_FILE.tmp"
    mv "$SENTENCES_FILE.tmp" "$SENTENCES_FILE"
    echo "Frases limitadas a $MAX_ENTRIES"
fi

# Limitar pares de frases se necessário
total=$(wc -l < "$SENTENCES_PAIRS_FILE")
if [ $total -gt $MAX_ENTRIES ]
then
    head -n $MAX_ENTRIES "$SENTENCES_PAIRS_FILE" > "$SENTENCES_PAIRS_FILE.tmp"
    mv "$SENTENCES_PAIRS_FILE.tmp" "$SENTENCES_PAIRS_FILE"
    echo "Pares de frases limitados a $MAX_ENTRIES"
fi

echo ""
echo "Resumo final:"
echo "  Palavras: $(wc -l < "$WORDS_FILE")"
echo "  Pares de palavras: $(wc -l < "$WORDS_PAIRS_FILE")"
echo "  Frases: $(wc -l < "$SENTENCES_FILE")"
echo "  Pares de frases: $(wc -l < "$SENTENCES_PAIRS_FILE")"
```

2.10.1 Output esperado

```
francisco@francisco-VirtualBox:~/Sistemas-Operativos-TG1/tg1/scripts$ ./limit_sentences_req10.sh
Entradas atuais em sentences.txt: 34558
Nothing to do: já tem <= 250000 entradas.
francisco@francisco-VirtualBox:~/Sistemas-Operativos-TG1/tg1/scripts$
```

2.11 Script Opcional (Mestre)

O script **master.sh** funciona como um menu que permite ao utilizador executar facilmente todos os scripts do trabalho, sem ter de os correr um a um manualmente. Quando o script é iniciado, ele mostra uma lista numerada com todas as opções, cada uma correspondente a um requisito do TG1.

O utilizador escolhe um número, e o *master.sh* executa automaticamente o script correto. Depois de cada execução, o programa pausa e espera que o utilizador pressione ENTER, para que consiga ler as mensagens antes de continuar.

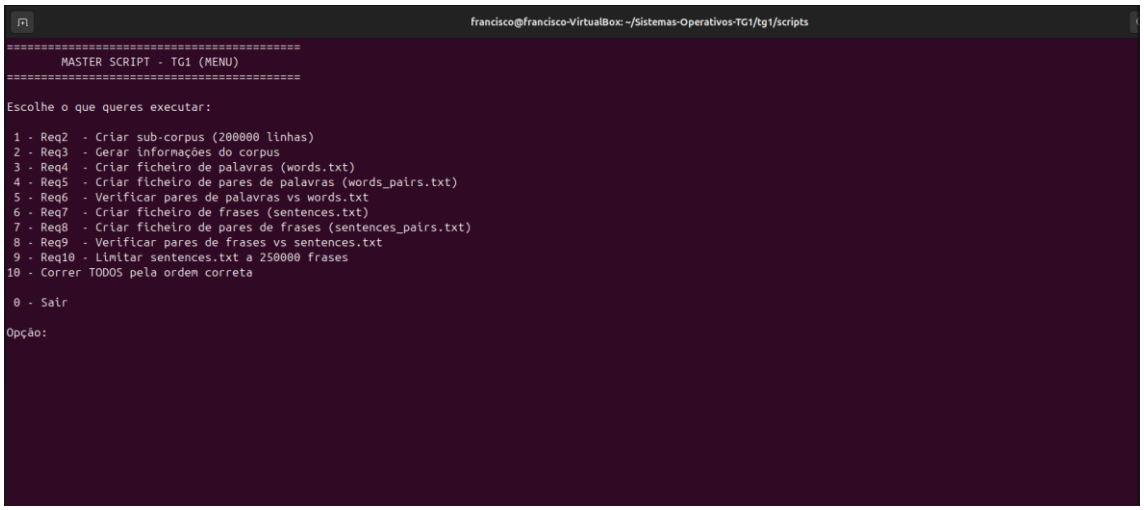
O script também inclui uma opção especial que prepara os ficheiros finais para instalar no software Eugénio. Esta opção copia os dicionários criados, muda-lhes o nome para os ficheiros que o Eugénio usa (geral.pal, geral.par, etc.) e coloca tudo numa pasta própria.

Existe ainda uma opção que executa **todos os scripts pela ordem certa**, desde o início até ao fim, garantindo que tudo é feito corretamente.

Em resumo, o **master.sh** serve para:

- facilitar a execução dos requisitos;
- evitar erros ao correr scripts manualmente;
- preparar automaticamente os ficheiros finais para o Eugénio;
- organizar todo o processo de forma simples e rápida.

2.11.1 Output esperado



2.12 Copiador de ficheiros

Por fim foi pedido que os alunos criassem um script Windows que copiasse os principais ficheiros gerados pelos scripts mencionados anteriormente para os seus respetivos ficheiros no software Eugénio. O objetivo deste requisito é que o Eugénio possa entender e sugerir palavras e frases em italiano ao utilizador.
Para tal foi criado um script no formato ".bat" visto ser o suportado pelo sistema operativo "Windows". O script começa por armazenar o caminho dos ficheiros a copiar em variável, de seguida verifica se o utilizador instalou a versão 64 bits ou 32 e então copia os ficheiros anteriormente guardados em variáveis para os que lhes correspondem na pasta do Eugénio. Há que ter em conta que de maneira ao script funcionar corretamente o utilizador deve substituir o caminho da variável "SOURCE_DIR"pelo caminho para a pasta onde os outputs gerados anteriormente estão.

```
@echo off
REM ============================================================
REM  Instala dicionários Italianos no Eugénio 3.0
REM  Copia os ficheiros já prontos da pasta C:\Eugenio_IT
REM  para:
REM    - pasta de instalação (Program Files)
REM    - pasta do utilizador (AppData\Roaming)
REM ============================================================

chcp 65001 >NUL

REM Pasta onde estão os ficheiros já prontos
set "SOURCE_DIR=C:\Eugenio_IT"

REM ------------------------------
REM Detectar diretoria do Eugénio
REM ------------------------------
if exist "C:\Program Files\Eugénio" (
    set "TARGET_DIR=C:\Program Files\Eugénio"
) else if exist "C:\Program Files (x86)\Eugénio" (
    set "TARGET_DIR=C:\Program Files (x86)\Eugénio"
) else (
    echo ERRO: O Eugénio nao foi encontrado.
    pause
    exit /b
)

REM Diretoria AppData do utilizador
set "USER_DIR=%AppData%\LabSI2-INESC-ID\Eugénio 3.0"

echo.
echo A copiar ficheiros...
echo.

REM Copiar para Program Files
copy /Y "%SOURCE_DIR%\*" "%TARGET_DIR%"

REM Criar pasta AppData caso não exista
if not exist "%USER_DIR%" mkdir "%USER_DIR%"

REM Copiar para AppData
copy /Y "%SOURCE_DIR%\*" "%USER_DIR%"

echo.
echo Ficheiros copiados:
echo      -> %TARGET_DIR%
echo      -> %USER_DIR%
echo.
pause
```

2.12.1 Output esperado

Capítulo 3

Conclusão

A realização deste trabalho permitiu desenvolver um conjunto completo de scripts em bash capazes de transformar um corpus de texto italiano nos dicionários necessários para o funcionamento do software Eugénio V3. Ao longo do projeto foram aplicados conceitos essenciais de sistemas operativos, como automação, manipulação de ficheiros, uso de comandos Unix e criação de pipelines para processamento eficiente de dados.

Todos os requisitos definidos no enunciado foram cumpridos: desde a criação do sub-corpus, à análise estatística do texto, à geração dos ficheiros de palavras, frases e respetivos pares, até à verificação de consistência e limitação de entradas para garantir compatibilidade com o Eugénio. A implementação do **Master Script** permitiu ainda integrar todo o processo num sistema simples e automático, facilitando a execução sequencial de todos os passos.

O trabalho contribuiu para uma compreensão prática sobre como construir dicionários linguísticos a partir de grandes volumes de texto, reforçando competências em automação, análise de dados e programação em shell. No final, foi possível gerar corretamente os ficheiros finais em italiano, prontos para instalação e utilização no sistema Eugénio V3.