Universidade do Minho
Licenciatura em Ciências da Computação

Processamento de Linguagens e Compiladores

Trabalho Prático 1

BibTeXPro, Um processador de BibTeX

Grupo 13

Pedro Faria
A72640

João Gouveia
A87995

João Goulart
A82643

Novembro 2021

Conteúdo

1 Introdução

No âmbito da Unidade Curricular de Processamento de Linguagens e Compiladores, desenvolvemos o projeto **BibTeXPro, Um processador de BibTeX** que propõe a implementação de um processador capaz de filtrar todas as entradas de uma base de dados textual. Este documento encontra-se estruturado de forma a que seja feita a análise de cada uma das tarefas executadas, bem como os métodos e abordagens que o nosso grupo utilizou para as executar, seguida de exemplos de funcionamento (incluindo Inputs e respectivos Outputs)

Como forma de familiarização com o formato BibTeX foi sugerida a consulta do ficheiro exemplo-utf8.bib. É-nos ainda explicado que as categorias de referência são identificadas pela primeira palavra logo após o caracter "@". Assim sendo, começamos por efetuar a **Análise do documento BibTeX** acima referido.

2 Descrição da abordagem seguida

Após analisar o documento BibTex, procedemos para a realização das tarefas que devíamos executar. Abordaremos, então, cada uma destas tarefas, explicando todos os pensamentos que tivemos e detalhes do seu desenvolvimento.

a) Contagem das categorias que ocorrem no documento

Tendo em mente que as categorias de referência são identificadas pela primeira palavra após o carácter "@", rapidamente estabelecemos um método para a identificação das categorias presentes no documento BibTeX. Para tal, formulámos uma expressão regular que nos permitisse encontrar todas as palavras do documento que se encontrassem logo a seguir a um carácter "@". Chegámos, então, à seguinte Expressão Regular:

```
1 pattern\_categorias = r'(@[\w]+{(.+\n)+}\n\n)'
```

Para nos auxiliar em algumas das alíneas seguintes à a), utilizámos um dicionário *dict_cat*, que tem o nome da categoria (*string*) como chave, como valor tem uma lista de dois elementos cujo primeiro elemento é a incidência (*integer*) e o segundo um dicionário com a chave da entrada como chave (*string*) e valor uma lista de dois elementos, cujo primeiro elemento é uma lista dos autores e o segundo é o título da entrada. Mas por enquanto, só nos iremos focar na incidência.

```
    dict_cat = dict()
    for categoria in categorias:
    categoria = categoria[0]
    categoria_nome = re.findall(pattern_nome_cat,categoria)[0].lower()
    #(...)
    incidencia = 1
    #(...)
    if categoria_nome in dict_cat:
        incidencia += dict_cat[categoria_nome][0]
    #(...)
    dict_cat[categoria_nome] = [incidencia,entradas]
```

Como podemos observar no excerto de código mostrado, a variável de incidência é inicializada com valor 1. Caso esta categoria já exista no dicionário, é adicionado o valor já presente à variável de incidência, de forma a atualizá-la.

b) Filtrar a chave, os autores e o título associados a cada categoria

Utilizando o mesmo método da alínea a), definimos uma Expressão Regular que nos auxiliasse a filtrar apenas a chave, os autores e o título de cada categoria. Para esse efeito, utilizamos as Expressões Regulares

```
pattern_chave = r'(?<={)[\w:-]+(?=,\n)'
pattern_autor = r'author *= *({|"){*(([\w.\s]|,|-|\\\'|\\\~)+)(?=}|",\n)'
pattern_titulos = r' title *= *({|")((.*)|(.*\n.*)|(\n.*))(}|")'
```

para filtrar as chaves, autores e títulos, respectivamente.
Estas expressões foram utilizadas de modo a conseguirmos extrair os dados para o dicionário criado na alínea a).

```
    for categoria in categorias:
    #(...)
    chave = re.findall(pattern_chave,categoria)[0]
    autor = re.findall(pattern_autor,categoria, re.IGNORECASE)[-1][1]
    autores = re.split(r' *and +| +and\n *',autor,100,re.IGNORECASE)
    autores = addAutor(dict_aut, autores)
    #(...)
    entradas = dict()

    try:
      titulo = re.findall(pattern_titulos,categoria, re.IGNORECASE)[0]
```

```
    except:
      continue

    if categoria_nome in dict_cat:
        #(...)
        entradas = dict_cat[categoria_nome][1]

    entradas[chave] = [autores,titulo]
    dict_cat[categoria_nome] = [incidencia,entradas]
```

Usamos duas estratégias adicionais para realizarmos esta extração de dados:

- a função *addAutor*, que foi criada de forma a separar os vários autores extraídos com a expressão regular e guardá-los num dicionário (sendo a chave o nome de um autor e o valor os outros autores com quem ele trabalhou) e numa lista.

```
    def addAutor(dic_aut,autores):
        for x in range(len(autores)):
            autores[x] = re.split(r', ',autores[x],re.IGNORECASE)
            if len(autores[x]) == 2:
                autores[x] = autores[x][1] + ' ' + autores[x][0]
            else:
                autores[x] = autores[x][0]

        if '' in autores:
            autores.remove('')

        for autor1 in autores:
            for autor2 in autores:
                autor1 = ' '.join(re.split(r'\n+ *',autor1,re.IGNORECASE))
                autor2 = ' '.join(re.split(r'\n+ *',autor2,re.IGNORECASE))
                if autor1 != autor2:
                    if autor1 not in dic_aut:
                        dic_aut[autor1] = list()
                    dic_aut[autor1].append(autor2)

        return autores
```

- lidar com uma exceção relacionada com a expressão regular dos títulos, que não funcionava para apenas uma entrada no ficheiro de exemplos.

```
    try:
      titulo = re.findall(pattern_titulos,categoria, re.IGNORECASE)[0]
```

```
    except:
      continue
```

Depois de obtermos o dicionário *dict_cat* completo, entregamo-lo como input na função *html_builder*() que estrutura o ficheiro HTML.
A nossa decisão final foi criar uma única tabela que engloba as alíneas a) e b). A tabela contém duas filas com duas colunas para a informação das categorias pedida na alínea a) e imediatamente a baixo tem todos os livros relacionados com a categoria especifica, organizados por chave, autores e título.

```
#(...)
for categoria in dict_cat:
  html += '''
      <tr>
        <th class="bortop">Categoria</th>
        <td class="tabcen bortop" colspan="2">{0}</td>
      </tr>'''.format(categoria)
  html += '''
      <tr>
        <th class="borbot">Incidencia</th>
        <td class="tabcen borbot" colspan="2">{0}</td>
      </tr>'''.format(str(dict_cat[categoria][0]))
  html += '''
      <tr>
        <th>Chave</th>
        <th>Autores</th>
        <th>Titulos</th>
      </tr>
#(...)
```

Escrevemos também uma pequena porção de código CSS para a tabela ser de mais fácil leitura.

```
.tabcen {
      text-align: center;
    }
      td,th {
      border: 1px solid grey;
    }
    th {
      background-color: #80808054;
    }
```

```
        .bortop {
          border-top: 5px solid black;
        }
        .borbot {
          border-bottom: 2px solid black;
        }
```

c) Criar um índice de autores, mapeando cada autor nos respectivos registos

Para realizarmos esta alínea, também utilizamos o mesmo ciclo utilizado nas alíneas anteriores, aproveitando também a lista de autores e as informações já retiradas (chave, titulo, nome da categoria).

```
    for autor in autores:
      if autor.lstrip() not in dict_aut_indice:
        dict_aut_indice[autor.lstrip()] = list()
      dict_aut_indice[autor.lstrip()].append([chave,titulo[1],categoria_nome])
```

Em seguida, utilizamos a informação adicionada ao dicionário para escrever uma *string* ordenada por autor alfabeticamente organizada da seguinte forma:

```
    Nome de autor (x entradas):
        "Título de livro 1", chave, nome da categoria
        "Título de livro 2", chave, nome da categoria
        ...
        "Título de livro x", chave, nome da categoria
```

d) Construção de um grafo que mostre todos os autores que publicaram com um dado autor

Usando o dicionário *dict_aut* definido na alínea b), definimos a seguinte função que constroi um grafo em DOT com as relações entre autores.
(Consideramos como autores que colaboram frequentemente autores que tivessem escrito mais que dois livros juntos.)

```
1   def graph_builder(dict_aut,autor):
2     for autor1 in dict_aut:
3       dict_aut_temp = dict_aut[autor1]
4       for autor2 in dict_aut_temp:
5         if dict_aut[autor1].count(autor2) < 2:
6           dict_aut[autor1].remove(autor2)
7       dict_aut[autor1] = set(dict_aut[autor1])
8
9     graph_dot = "digraph G {\n"
10
11    if autor:
12      for autor2 in dict_aut[autor]:
13          graph_dot += '"{0}" -> "{1}"\n'.format(autor,autor2)
14    else:
15      for autor1 in dict_aut:
16        for autor2 in dict_aut[autor1]:
17          graph_dot += '"{0}" -> "{1}"\n'.format(autor1,autor2)
18
19    graph_dot += "}"
20
21    return graph_dot
```

Caso não tenha sido fornecido nenhum imput na consola, irá ser gerado um grafo com todos os autores. Se for fornecido o input, o código irá só construir o grafo a partir do autor indicado.

3 Código

Abaixo encontra-se o código python do nosso trabalho na sua totalidade.

```
#!/usr/bin/python
# -*- coding: utf8 -*-

import re
from autor_builder import autor_builder
from html_builder import html_builder
from indice_builder import indice_builder
from graph_builder import graph_builder

f = open('exemplo-utf8.bib','r', encoding="utf8")

pattern_categorias = r'(@[\w]+{(.+\n)+}\n\n)'
pattern_nome_cat = r'(?<=@)[\w]+(?={)'
pattern_chave = r'(?<={)[\w:-]+(?=,\n)'
pattern_autor = r'author *= *({|"){*(([\w.\s]|,|-|\\\'|\\\~)+)(?=}|",\n)'
pattern_titulos = r' title *= *({|")((.*)|(.*\n.*)|(\n.*))(}|")'

categorias = re.findall(pattern_categorias,f.read())
f.close()
dict_aut = dict()
dict_cat = dict()
dict_aut_indice = dict()
#dict_cat {cat: [incidencia, {chave:[[autores],titulo]}]}

for categoria in categorias:
    categoria = categoria[0]
    categoria_nome = re.findall(pattern_nome_cat,categoria)[0].lower()

    chave = re.findall(pattern_chave,categoria)[0]
    autor = re.findall(pattern_autor,categoria, re.IGNORECASE)[-1][1]
    autores = re.split(r' *and +| +and\n *',autor,100,re.IGNORECASE)
    autores = autor_builder(dict_aut, autores)
    incidencia = 1
    entradas = dict()

    try:
      titulo = re.findall(pattern_titulos,categoria, re.IGNORECASE)[0]
    except:
      continue

    if categoria_nome in dict_cat:
```

```
        incidencia += dict_cat[categoria_nome][0]
        entradas = dict_cat[categoria_nome][1]

    entradas[chave] = [autores,titulo]
    dict_cat[categoria_nome] = [incidencia,entradas]

    for autor in autores:
      if autor.lstrip() not in dict_aut_indice:
        dict_aut_indice[autor.lstrip()] = list()
      dict_aut_indice[autor.lstrip()].append([chave,titulo[1],categoria_nome])

html = html_builder(dict_cat)
open('index.html','w').write(html)
print("Ficheiro HTML criado com sucesso.")

indice_autores = indice_builder(dict_aut_indice)
open('indice_autores.txt','w').write(indice_autores)
print("Ficheiro TXT criado com sucesso.")

autor = input('Autor: ')
graph = graph_builder(dict_aut,autor)
open('graph.dot','w').write(graph)
print("Ficheiro DOT criado com sucesso.")
```

4 Exemplos de utilização

Para demonstrar o funcionamento do nosso trabalho introduzimos o seguinte conteúdo como input para o nosso programa.

```
@techreport{Camila,
  author ={{projecto Camila}},
  editor ={L.S. Barbosa and J.J. Almeida and J.N. Oliveira and Luís Neves},
  title = "\textsc{Camila} - A Platform for Software Mathematical Development",
  url="http://camila.di.uminho.pt",
  type="(Páginas do projecto)",
  institution = "umdi",
  year=1998,
  keyword = "FS",
}

@techreport{Barbosa95b,
   author = "L.S. Barbosa and J.J. Almeida",
   title = "Growing Up With \textsc{Camila}",
   institution =   "umdi",
   year   = 1995,
   number = "DI-CAM-95:7:1",
   url = "http://www.di.uminho.pt/~lsb/pub_camila/romantic.ps.gz",
 keyword ="Camila, formal specification, didatics",
}

@inproceedings{Ramalho95,
   author = "J.C. Ramalho and J.J. Almeida and P.R. Henriques",
   title = "Algebraic Specification of Documents",
   booktitle = "TWLT10 - Algebraic Methods in Language Processing",
   year = 1995,
   month = "6--8 Dec.",
   editor = "A. Nijholt and G. Scollo and R. Steetskamp",
   address = "Twente University, Netherlands",
   note = "AMiLP'95",
   series = "Twente Workshop on Language Technology",
   url="http://natura.di.uminho.pt/~jj/bib/amilp95.ps.gz",
   docpage="http://www.di.uminho.pt/~jcr/projectos/david/ARTIGOS/AMiLP95/amilp95.html",
   pages = "55--64",
   keyword ="PDavid, Camila, SGML",
}

@article{sepln06,
  author =       {Alberto Simões and J. João Almeida},
  title =        {{NatServer:} A Client-Server Architecture for building Parallel
```

```
  Corpora applications},
  year =         {2006},
  journal =    {Procesamiento del Lenguaje Natural},
  address =      {Zaragoza, Spain},
  url = {http://alfarrabio.di.uminho.pt/~albie/publications/sepln06.pdf},
  month =        {September},
  lang =         {EN},
  volume =       {37},
  pages =        {91--97},
  abstract =     {Parallel corpora are important resources for most
                  Natural Language processing tasks. From the common
                  applications, like machine translation, to the
                  usually mono-lingual tasks as paraphrase detection
                  and word sense disambiguation, most researchers are
                  using massive parallel corpora.  Thus, the
                  availability of an efficient way to manage them is
                  very important.  This paper presents a Client-Server
                  architecture to query efficiently parallel corpora
                  and probabilistic translation dictionaries.},
}

@Article{MSH05,
  author = {Giovana Mendes and Nuno Alberto Silva  and Pedro Rangel Henriques},
  title = {Utilizando uma Base de Dados XML Nativa aplicada ao tratamento de
  erros num sistema de logs},
  journal = {Sistemas de Informação},
  editor = {},
  publisher = {APSI: Associação Portuguesa de Sistemas de Informação},
  year = {2005},
  month = {},
  volume = {17},
  number = {},
  pages = {91-100}
}

@Article{ALHF02,
  author = {Gustavo Arnold and Giovani Librelotto  and Pedro Rangel Henriques
  and Jaime Fonseca},
  title = {O Uso da Linguagem RS em Robótica},
  journal = {Revista Electrónica e Telecomunicações},
  editor = {},
  publisher = {Departamento de Electrónica e Telecomunicações da Universidade de Aveiro},
  year = {2002},
  month = {Apr},
```

```
  volume = {},
  number = {},
  pages = {501-508}
}
```

O código/ficheiro HTML para responder às alíneas a) e b) gerado foram os seguintes.

```
<!DOCTYPE html>
<html>
  <head>
    <style>
      .tabcen {
        text-align: center;
      }
        td,th {
        border: 1px solid grey;
      }
      th {
        background-color: #80808054;
      }
      .bortop {
        border-top: 5px solid black;
      }
      .borbot {
        border-bottom: 2px solid black;
      }
    </style>
  </head>
  <body>
    <table>
      <tbody>
        <tr>
          <th class="bortop">Categoria</th>
          <td class="tabcen bortop" colspan="2">techreport</td>
        </tr>
        <tr>
          <th class="borbot">Incidencia</th>
          <td class="tabcen borbot" colspan="2">2</td>
        </tr>
        <tr>
          <th>Chave</th>
          <th>Autores</th>
          <th>Titulos</th>
        </tr>

        <tr>
          <td>Camila</td>
          <td>
```

```
            <ul>
              <li>projecto Camila</li>
            </ul></td>
            <td>\textsc{Camila} - A Platform for Software Mathematical Development</td>
          <tr>
            <td>Barbosa95b</td>
            <td>
            <ul>
              <li>L.S. Barbosa</li>
              <li>J.J. Almeida</li>
            </ul></td>
            <td>Growing Up With \textsc{Camila}</td>
          <tr>
            <th class="bortop">Categoria</th>
            <td class="tabcen bortop" colspan="2">inproceedings</td>
          </tr>
          <tr>
            <th class="borbot">Incidencia</th>
            <td class="tabcen borbot" colspan="2">1</td>
          </tr>
          <tr>
            <th>Chave</th>
            <th>Autores</th>
            <th>Titulos</th>
          </tr>

          <tr>
            <td>Ramalho95</td>
            <td>
            <ul>
              <li>J.C. Ramalho</li>
              <li>J.J. Almeida</li>
              <li>P.R. Henriques</li>
            </ul></td>
            <td>Algebraic Specification of Documents</td>
          <tr>
            <th class="bortop">Categoria</th>
            <td class="tabcen bortop" colspan="2">article</td>
          </tr>
          <tr>
            <th class="borbot">Incidencia</th>
            <td class="tabcen borbot" colspan="2">3</td>
          </tr>
          <tr>
```

```
            <th>Chave</th>
            <th>Autores</th>
            <th>Titulos</th>
          </tr>

          <tr>
            <td>sepln06</td>
            <td>
            <ul>
              <li>Alberto Simões</li>
              <li>J. João Almeida</li>
            </ul></td>
            <td>{NatServer:} A Client-Server Architecture for building Parallel Corpora application
          <tr>
            <td>MSH05</td>
            <td>
            <ul>
              <li>Giovana Mendes</li>
              <li>Nuno Alberto Silva</li>
              <li>Pedro Rangel Henriques</li>
            </ul></td>
            <td>Utilizando uma Base de Dados XML Nativa aplicada ao tratamento de erros num sistema
          <tr>
            <td>ALHF02</td>
            <td>
            <ul>
              <li>Gustavo Arnold</li>
              <li>Giovani Librelotto</li>
              <li>Pedro Rangel Henriques</li>
              <li>Jaime Fonseca</li>
            </ul></td>
            <td>O Uso da Linguagem RS em Robótica</td>
      </tbody>
    </table>
  </body>
</html>
```

Categoria	techreport	
Incidencia	2	
Chave	**Autores**	**Titulos**
Camila	• projecto Camila	\textsc{Camila} - A Platform for Software Mathematical Development
Barbosa95b	• L.S. Barbosa • J.J. Almeida	Growing Up With \textsc{Camila}

Categoria	inproceedings	
Incidencia	1	
Chave	**Autores**	**Titulos**
Ramalho95	• J.C. Ramalho • J.J. Almeida • P.R. Henriques	Algebraic Specification of Documents

Categoria	article	
Incidencia	3	
Chave	**Autores**	**Titulos**
sepln06	• Alberto Simões • J. João Almeida	{NatServer:} A Client-Server Architecture for building Parallel Corpora applications
MSH05	• Giovana Mendes • Nuno Alberto Silva • Pedro Rangel Henriques	Utilizando uma Base de Dados XML Nativa aplicada ao tratamento de erros num sistema de logs
ALHF02	• Gustavo Arnold • Giovani Librelotto • Pedro Rangel Henriques • Jaime Fonseca	O Uso da Linguagem RS em Robótica

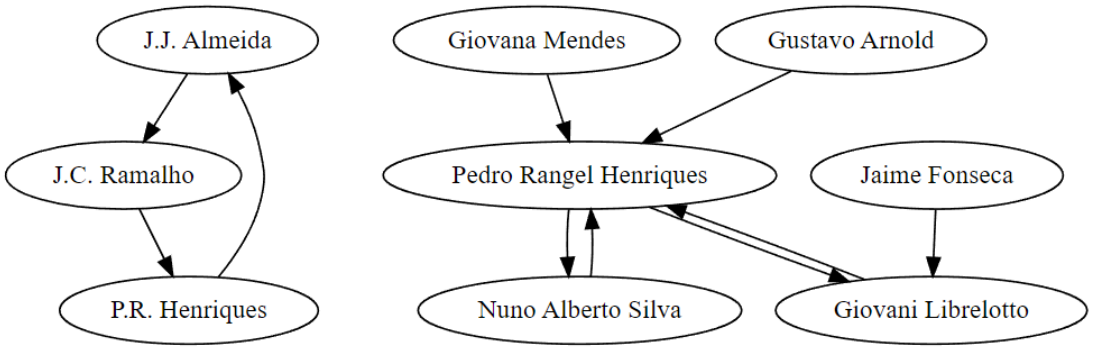
O ficheiro de texto com o índice dos autores gerado foi o seguinte.

```
A
Alberto Simões (1 entrada):
 "{NatServer:} A Client-Server Architecture for building Parallel Corpora
 applications", sepln06, article


G
Giovana Mendes (1 entrada):
 "Utilizando uma Base de Dados XML Nativa aplicada ao tratamento de erros num
 sistema de logs", MSH05, article


Giovani Librelotto (1 entrada):
 "O Uso da Linguagem RS em Robótica", ALHF02, article


Gustavo Arnold (1 entrada):
 "O Uso da Linguagem RS em Robótica", ALHF02, article


J
J. João Almeida (1 entrada):
 "{NatServer:} A Client-Server Architecture for building Parallel Corpora
 applications", sepln06, article


J.C. Ramalho (1 entrada):
 "Algebraic Specification of Documents", Ramalho95, inproceedings


J.J. Almeida (2 entradas):
 "Growing Up With \textsc{Camila}", Barbosa95b, techreport
 "Algebraic Specification of Documents", Ramalho95, inproceedings


Jaime Fonseca (1 entrada):
 "O Uso da Linguagem RS em Robótica", ALHF02, article


L
L.S. Barbosa (1 entrada):
 "Growing Up With \textsc{Camila}", Barbosa95b, techreport
```

```
43
44
45  N
46  Nuno Alberto Silva (1 entrada):
47   "Utilizando uma Base de Dados XML Nativa aplicada ao tratamento de erros num
48   sistema de logs", MSH05, article
49
50
51  P
52  P.R. Henriques (1 entrada):
53   "Algebraic Specification of Documents", Ramalho95, inproceedings
54
55
56  Pedro Rangel Henriques (2 entradas):
57   "Utilizando uma Base de Dados XML Nativa aplicada ao tratamento de erros num
58   sistema de logs", MSH05, article
59   "O Uso da Linguagem RS em Robótica", ALHF02, article
60
61
62  projecto Camila (1 entrada):
63   "\textsc{Camila} - A Platform for Software Mathematical Development", Camila, techreport
```

Em seguinda, mostramos o código do grafo gerado sem introduzir o nome de nenhum autor e a sua representação gráfica; e o código do grafo gerado se for introduzido o nome "Pedro Rangel Henriques"e a sua representação gráfica.

```
digraph G {
"J.J. Almeida" -> "J.C. Ramalho"
"J.C. Ramalho" -> "P.R. Henriques"
"P.R. Henriques" -> "J.J. Almeida"
"Giovana Mendes" -> "Pedro Rangel Henriques"
"Nuno Alberto Silva" -> "Pedro Rangel Henriques"
"Pedro Rangel Henriques" -> "Giovani Librelotto"
"Pedro Rangel Henriques" -> "Nuno Alberto Silva"
"Gustavo Arnold" -> "Pedro Rangel Henriques"
"Giovani Librelotto" -> "Pedro Rangel Henriques"
"Jaime Fonseca" -> "Giovani Librelotto"
}
```




```
digraph G {
"Pedro Rangel Henriques" -> "Giovani Librelotto"
"Pedro Rangel Henriques" -> "Nuno Alberto Silva"
}
```