Processamento de Linguagens (3º Ano)

Trabalho Prático 3
(YACC)

Relatório de Desenvolvimento

Gonçalo Camaz (a76861)

José Ferreira (a78452)

Jorge Oliveira (a78660)

12 de Junho de 2018

Resumo

No presente relatório vamos apresentar todo o processo de desenvolvimento de uma linguagem que tem por objetivo a adição de músicas e ainda a apresentação das mesmas. Apresentaremos uma descrição informal do nosso problema e quais os nossos objetivos com a realização deste trabalho, e posteriormente, vamos mais ao pormenor do nosso desenvolvimento, explicando os algoritmos e estrutras envolvidas no nosso processo de criação da linguagem. Para a criação desta linguagem utilizamos o par *lex/yacc* e ainda a linguagem de programação *C*, *MySQL* para guardar os dados persistentemente e ainda *graphviz* para o desenho de grafos.

Conteúdo

Capítulo 1

Introdução

Neste terceiro trabalho o grande objetivo era desenvolver uma linguagem segundo o método da tradução dirigida pela sintaxe, suportado numa gramática tradutora utilizando como ferramentas o par lex/yacc e ainda a linguagem imperativa C.

O presente relatório serve para descrever todo o processo para a criação de uma linguagem, que tem como objetivo descrever uma música e posteriormente adicioná-la aos dados. Posteriormente é ainda feito um pedido, através da indicação do estado de espírito do utilizador, e são escolhidas as músicas que mais se adequam ao estado de espírito.

Como seria de esperar, o nosso principal objetivo era conseguir escrever esta linguagem segundo os conhecimentos apreendidos e as restrições definidas por parte do enunciado. Para além deste tínhamos também como o objetivo a construção de uma base de dados que tivesse a capacidade para guardar toda a informação, bem como adicionar todas as estruturas determinantes, na linguagem C, para poder aceder à informação requerida. Por último, para a apresentação dos resultados, tínhamos como objetivo a representação na forma de uma grafo circular, que permite ao utilizador ter uma melhor noção de quais as músicas que se aproximam mais do seu estado espírito, indicando anteriormente.

Neste trabalho, foi ainda deixado ao critério, a forma como nós nos tínhamos de fazer a aproximação do estado de espírito consoante os tipos de sentimentos presentes nas músicas, optando nós por dar ao utilizador sempre uma música que se aproximasse mais do seu estado de espírito. E não tentando, por exemplo, espevitar o utilizador, quando este nos indicasse que se sentia triste.

Estrutura do Relatório

No capítulo 2 será feita uma apresentação do problema que estamos a tratar e ainda os objetivos traçados pelo grupo de trabalho. Desta forma o leitor terá um melhor enquadramento sobre o porquê de realizarmos este trabalho.

No capítulo 3 apresentaremos a forma como abordamos o problema em termos algorítmicos e das estruturas codificads e no capítulo 4 serão apresentados os resultados obtidos pelo nosso programa. O capítulo 5 concluí o relatório com a apresentação da conclusão, onde abordaremos a experiência vivida duranto o processo de criação e uma análise critica do trabalho desenvolvido.

Capítulo 2

Análise e Especificação

2.1 Descrição informal do problema

A nossa gramática, tal como já foi sucintamente explicado anteriormente, tem como objetivo o carregamento de músicas para a nossa "Biblioteca"e posteriormente a efetuação de um pedido de quais as músicas que mais se adequam ao estado de espírito do utilizador, validando se a sintaxe de escrita, tanto do carregamento como do pedido, se encontram corretos.

Era necessário definir o método pelo qual se efetuava a seleção das músicas baseado no estao de espirito do utilizador. Sempre tendo em atenção que na nossa lingua portuguesa existem sinónimos. Por exemplo, o utilizador poderá indicar que está calmo, porém se uma música passar uma sensação de relaxamento, esta música também poderá ser indicada para o utilizador que se encontra calmo.

Para a apresentação dos resultados, decidimos que o melhor seria com a visualização de um grafo circular. Já que temos uma noção imediata de que as músicas que se encontram mais perto do centro são as que mais se adequam mais ao nosso estado de espírito.

2.2 Objetivos

A definição dos objetivos já foi abordada um pouco na introdução, porém nesta secção faremos uma sintese dos mesmos:

- Escrever uma gramática seguindo as restrições impostas inicialmente
- A nossa gramática deverá ser capaz de validar o registo de músicas e pedidos das mesmas
- Efetuar um algoritmo de seleção que permita ao utilizador obter de uma forma eficiente as músicas que mais se adequam ao seu estado de espirito
- Utilizar a linguagem em C para manipular toda a lógica envolvida
- Utilizar todas as funcionalidades oferecidas pelo par lex/yacc para escrever a gramática tradutora
- Apresentar os resultados com um aspeto mais "user-friendly"e intuitivo

Capítulo 3

Concepção/desenho da Resolução

3.1 Gramática

Inicialmente necessitamos de definir uma gramática para reconhecer as músicas para a nossa **Musiteca** e para as guardar na Base de Dados. Para implementar a gramática efetuamos uma análise ao nosso problema para perceber o que definiria cada Música Espiritual. Após esta análise chegamos à seguinte conclusão: cada música é constituída pelos os seguintes campos:

- id - o id da música
- Nome - o nome da música
- Autores - o conjunto dos autores
- Ano - o ano de lançamento
- Editor - a editora
- Letra - a letra da música Decidimos por um *url* visto que existem letras online
- Estilo - o estilo da música
- Tipo - uma lista de pares que identificam os tipos espirituais da música. Os pares são constituídos por:

  - Descritor - o nome do estado de espírito
  - Percentagem - a percentagem do estado

  De realçar que a soma das percentagens terá de ser 100%.

- Video - um url para o vídeo, para que possa ser clicado e posto a tocar.

Após a definição do que é uma Música Espiritual e os seus constituintes decidimos implementar a gramátca correspondente.

```
Musicas: Musica ';' Musicas
  | Musica
;

Musica: ID':' Nome '|' Autores '|' Ano '|' Editor '|' Letra '|' Estilo '|' Tipo '|' Video

;

Nome: string Nome
| string
;

Autores: Autor
| Autor ';' Autores
;

Autor: string Autor
| string
;

Editor: string Editor
| string
;

Letra: url
;

Ano: num
;

Estilo: string
;

Tipo: '{' Descritor ',' Percentagem '}' ';' Tipo
```

```
| '{' Descritor ',' Percentagem '}'

;

Descritor: string
;

Percentagem: num
;

Video: url
;

ID: num
;
```

Decidimos utilizar recursividade à direita por nos parecer mais intuitivo e para cada um dos componentes temos sempre carateres entre os mesmos para que se possa identificar corretamente o fim de um e o início de outro. Para além disso podemos verificar que criamos 3 símbolos terminais:

- string - conjunto de letras
- num - números não negativos
- url - um conjnto de carateres que cmpõem um url (terá de ter no mínimo um '.' para não entrar em conflitos com o primeiro)

Por forma a colocar ações semânticas na gramática para realizar o pretendido decidimos criar algumas estruturas. Uma delas foi a correspondente a uma Música Espiritual e a lista dos estados de espírito.

```
/* Estruturas relevantes para a linguagem e para as querys */
typedef struct sTipo{
char* descritor; // nome do tipo
float percentagem; //percentagem desse tipo na musica
} TipoS;

typedef TipoS* Tipo;

typedef struct lTipo{
Tipo tipo;
struct lTipo* proximo;
```

```
} TipoL;

typedef TipoL* listaTipo;

typedef struct sMusica{
int id;
char* nome; // nome da musica
char* autores; // autores da musica
int ano; // ano da musica
char* editor; // editores da musica
char* letra; // letra da musica (pode ser um ficheiro com a letra ou um url também)
char* estilo; // estilo da musica
char* url; // url do video da musica
listaTipo tipos; // lista dos tipos da musica
} MusicaS;

typedef MusicaS* MEspiritual;
```

Para além disso definimos a union correspondente ao yylval.

```
%union {
float n;
char* c;
listaTipo tipos;
MEspiritual musica;
}
```

Após termos definido o que foi referido anteriormente e por forma a estabelecer as ações semânticas tivemos de estipular cada um dos Símbolos (Terminais e Não-Terminais), tendo delineado o seguinte:

- MEspiritual - Musica
- listaTipo - Tipo
- char* - Video Editor Estilo Autores Nome Autor Descritor Letra
- float - ID Ano Percentagem

De seguida implementamos as ações semâticas, sendo que as mesmas apenas fazem o seguinte:

1. Cada uma atualiza o seu valor
2. Nas referentes aos Descritor e Estilo verifica se na BD contém o valor correspondente (na tabela respetiva)
3. Verifica se a soma das percentagens de cada música é 100 %
4. Relativamente a cada Música constroi a MEspiritual correspondente e guarda-a na BD

Para além da linguagem que define as músicas decidimos também juntar à mesma, uma linguagem que define os pedidos. Cada Pedido é caraterizado por:

- Estado de espírito
- Percentagem desse estado

Para além disso podem existir 1 ou mais pedidos.

```
Pedidos: PedidoAux ';' Pedidos
| PedidoAux
;

PedidoAux: Pedido //criado para se poder realizar as ações semanticas separadamente
;

Pedido: '{' Descritor ',' Percentagem '}' ',' Pedido
| '{' Descritor ',' Percentagem '}'
;
```

As ações semãnticas desta parte são bastante simples, sendo que para cada Pedido são atualizadas as percentagens (caso contenha apenas 1 com 30% esse é considerado 100%) e de seguida cria a página com o Pedido.

A Musiteca é composta por Músicas seguidas de Pedidos.

```
Musiteca: Musicas '<' '/' '/' '>' Pedidos
;
```

3.1.1 Analisador Léxico

Por forma a reconhecer os símbolos terminais da Gramática definimos o seguinte analisador léxico.

```
%%

"|" {return yytext[0];}
";" {return yytext[0];}
"{" {return yytext[0];}
"}" {return yytext[0];}
":" {return yytext[0];}
"," { return yytext[0];}
"/" { return yytext[0];}
"<" { return yytext[0];}
">" { return yytext[0];}


[A-Za-z]+ { yylval.c = strdup(yytext); return string; }
[0-9]+(.[0-9]+)? { yylval.n = atof(yytext); return num; }
([A-Za-z0-9/:\-?=]+\.)+[A-Za-z0-9/:\-?=]+ { yylval.c = strdup(yytext); return url; }

.|\n {}
```

O mesmo reconhece cada um dos carateres que separa os diferentes Símbolos e retorna-os. Reconhece também os símbolos terminais referidos anteriormente colocando o seu valor na *union* e retornando o respetivo "código". De referir apenas que o **url** tem de ser constituído por pelo menos 1 ponto.

3.2 Estruturas auxiliares

Para poder facilitar o processo de criação dos pedidos decidimos utilizar algumas estruturas auxiliares, das quais:

- Uma árvore binária balanceada com 7 nodos que contêm
    1. Um valor correspondente ao limite superior da percentagem de acerto daquele nó

2. Um valor correspondente ao limite inferior da percentagem de acerto daquele nó
3. A lista de músicas que se encontram naquele intervalo de valores de acerto
4. O número de músicas presentes na lista

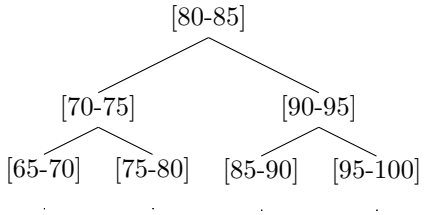
- A lista com as músicas que contém:
  1. A MEspiritual correspondente
  2. A percentagem de acerto

```
/* Estrutura para os nodos do grafo que se aproximam do que o utilizador quer */

/* Entre 65 e 100 por cento, em intervalos de 5% (7 intervalos) */
typedef struct musLista{
MEspiritual musica;
float aproximacao;
struct musLista* proxima;
} *listaMusicas;

typedef struct musArvore{

listaMusicas musicas;
float limiteInferior;
float limiteSuperior;
struct musArvore* esquerda;
struct musArvore* direita;
int numero;
} *arvoreMusicas;
```



3.3 Estruturas de Dados

3.3.1 Esquema Lógico da Base de Dados

Analisando atentamente o problema, vemos que facilmente, iríamos precisar de 2 tabelas para representar o nosso problema.

Uma é a tabela referente às caraterísticas gerais da música, 'Musica', que contém os seguintes atributos:

- **id -** que identifica unicamente uma música;
- **nome -** representa o nome da música;
- **autores -** identifica os cantores da música;
- **ano -** referente ao ano em que foi lançada;
- **editor -** identifica a editora que lançou a música;
- **letra -** guarda o link para o qual está presente a letra da música;
- **estilo -** identifica o género musical;
- **url -** guarda o link para o qual se pode pôr a reproduzir a música;

Como a música era caraterizada por diferentes sentimentos, então fez com que fosse determinante criar uma tabela para representar estes tipos de sentimentos. A tabela 'Tipo' é caraterizada pelos seguintes atributos:

- **descritor -** representa o sentimento da música;
- **percentagem -** representa qual a fatia predominante na música por parte do sentimento referido;
- **Musica -** é uma chave estrangeira para a tabela Musica;

Como os diferentes géneros musicais são limitados, tornou-se necessário para nós criar uma tabela onde guardámos quais os tipos que nós admitimos na nossa linguagem, que é feito através da tabela 'Estilo'.

Semelhante ao estilo musical, são também todas as sensações envolvidas na nossa linguagem. O conjunto de emoções é bastante diversificado, mas nós

limitamos este domínio, só deixando inserir certas emoções. Para mais uma vez guardar estas palavras, decidimos criar uma tabela, 'Descritores', que para além de guardar as emoções, as relacionava com um sinónimo aplicando-lhe uma percentagem.

3.3.2 Grafo

Para a geração do grafo utilizamos a API do graphviz, já que nos pareceu a que apresentava mais funcionalidades para o resultado que pretendiamos. Para tal arranjámos um template onde nos permitia inserir nodos de forma circular, para tal só tinhamos de definir uma posição, que foi feita através do uso de trignometria com a equação da circunferência.

Um dos atributos que poderíamos definir no nodo era o "URL"que foi utilizado para redirecionar para a página correspondente à música que o respetivo nodo representa.

Para a apresentação da música efetuamos uma simples tabela em *html* que nos permitia visualizar todos os atributos da música que guardamos na base de dados, sendo que dois deles são já uma hiperligação para o respetivo *url* e ainda qual o valor da percentagem de aproximação das caraterísticas da música consoante o pedido efetuado pelo utilizador.

No final, para a apresentação do grafo, criámos uma simples página em *html* que apenas faz a apresentação da imagem do grafo.

3.4 Algoritmos

3.4.1 Algoritmo de Seleção

O algoritmo de seleção é bastante simples, sendo que para cada Música que corresponde a um do estados de espírito do utilizador o que fazemos é o seguinte:

- Para cada um dos descritores do Tipo
  - Para cada um dos estados de espírito
    - Multiplica o valor da percentagem do Tipo na Música pelo valor da relação entre o Estado e o Tipo e pelo valor da percentagem

do Estado relativamente ao utilizador
* Soma o valor à percentagem total para a Música.

Capítulo 4

Codificação e Testes

4.1 Testes realizados e Resultados

Apresentaremos agora alguns testes realizados pelo grupo. Os dados para carregar as músicas e em seguida os pedidos são os seguintes:

```
1: God is Plan|Drake|2018|Cardo|
https://www.letras.mus.br/drake/gods-plan/|rap|{calmo,70};{dancante,30}|
https://www.youtube.com/watch?v=xpVfcZ0ZcFM ;
```

```
2: Chandelier|Sia|2014|Monkey Puzzle|
https://www.letras.mus.br/sia/chandelier/|pop|
{dancante,10};{espevitante,40};{agitado,50}|
https://www.youtube.com/watch?v=2vjPBrBU-TM ;
```

```
3: Dont worry be happy | Bob Marley | 1988 | Linda Goldstein |
https://www.letras.mus.br/drake/gods-plan/ | RandB | {alegre,100}
| https://www.youtube.com/watch?v=L3HQMbQAWRc ;
```

```
4: A chuva|Marisa|2014|Marisa|
https://www.letras.mus.br/mariza/485207/|fado|
{calmo,30};{relaxante,60};{triste,10}|
```

https://www.youtube.com/watch?v=tC88Oyz8Khs ;

5: Hallelujah|Leonard Cohen|1984|Leonard|
https://www.letras.mus.br/jeff-buckley/6098/|classico|{triste,30};{calmo,70}|
https://www.youtube.com/watch?v=YrLk4vdY28Q ;

6: Gangnam Style | Psy | 2012 | YG |
https://www.letras.mus.br/psy/gangnam-style/| pop |
{agitado,30};{alegre,30};{espevitante,40} |
https://www.youtube.com/watch?v=9bZkp7q19f0 ;

7: High Hopes | Kodaline | 2013 | B-Unique |
https://www.letras.mus.br/kodaline/high-hopes/traducao.html|rock|
{triste,60};{calmo,40}|
https://www.youtube.com/watch?v=E4povfmX144 ;

8: Let it Go|James Bay|2014|Republic|
https://www.letras.mus.br/james-bay/let-it-go/traducao.html|rock|
{calmo,50};{relaxante,50}|
https://www.youtube.com/watch?v=GsPq9mzFNGY ;

9: Reggaeton Lento|CNCO|2016|CNCO|
https://www.letras.mus.br/drake/gods-plan/|reggaeton|
{alegre,20};{dancante,50};{agitado,30}|
https://www.youtube.com/watch?v=7jpqqBX-Myw ;

10: Meant to be|Bebe Rexha|2017|Warner Bros|
https://www.letras.mus.br/bebe-rexha/meant-to-be/traducao.html|country|
{relaxante,100}| https://www.youtube.com/watch?v=zDo0H8Fm7d0 ;

11: A Maquina|Amor Eletro|2011|AE|
https://www.letras.mus.br/amor-electro/1881383/|rock|{triste,10};{calmo,90}|
https://www.youtube.com/watch?v=d1onT5Z51ZQ ;

12: A Vida Toda|Carolina Deslandes|2017|CD|
https://www.vagalume.com.br/carolina-deslandes/a-vida-toda.html|country|
{calmo,100}| https://www.youtube.com/watch?v=75iqd2yJH6w

<///>

{calmo,90},{relaxante,10};

{alegre,100}

Os resultados obtidos no primeiro e segundo pedido foram:

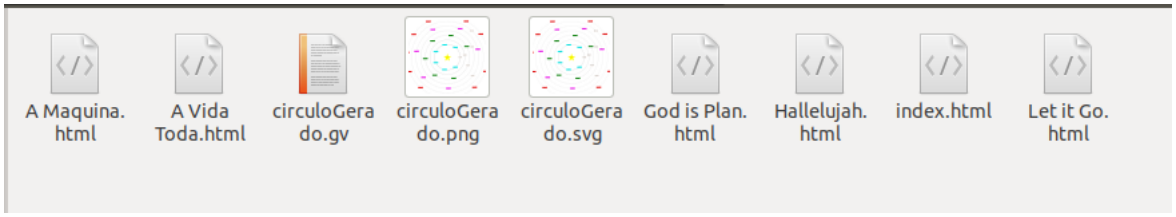


Figura 4.1: Ficheiros presentes na pasta criada do pedido 1

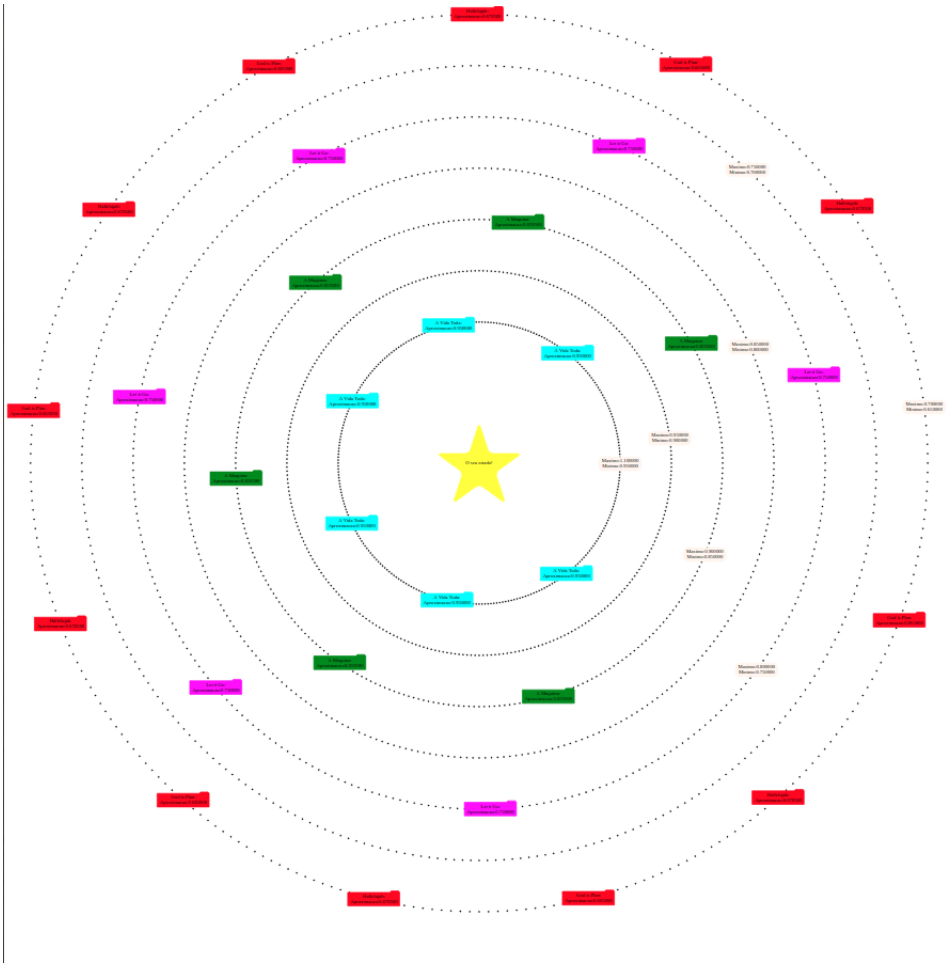


Figura 4.2: Grafo Total Gerado

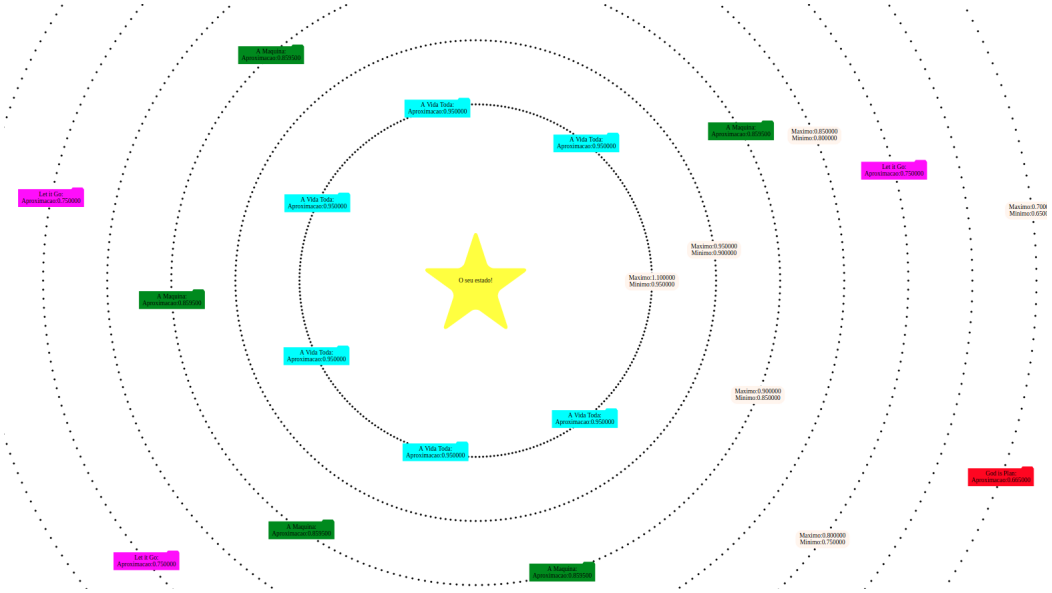


Figura 4.3: Zoom do grafo gerado

Informações da Música

Nome:	A Vida Toda
Autores:	Carolina Deslandes
Ano Lançamento:	2017
Editora:	CD
Letra:	Clique Aqui!
Estilo:	country
Letra:	Play!
Percentagem Acerto:	0.950000

Figura 4.4: Apresentação de uma música selecionada

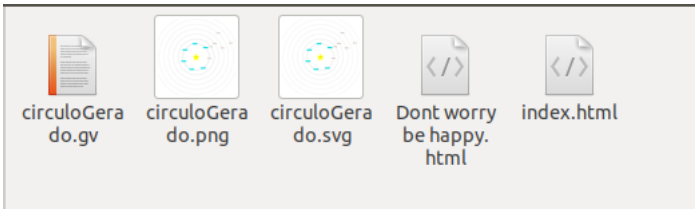


Figura 4.5: Ficheiros Gerados do Pedido 2

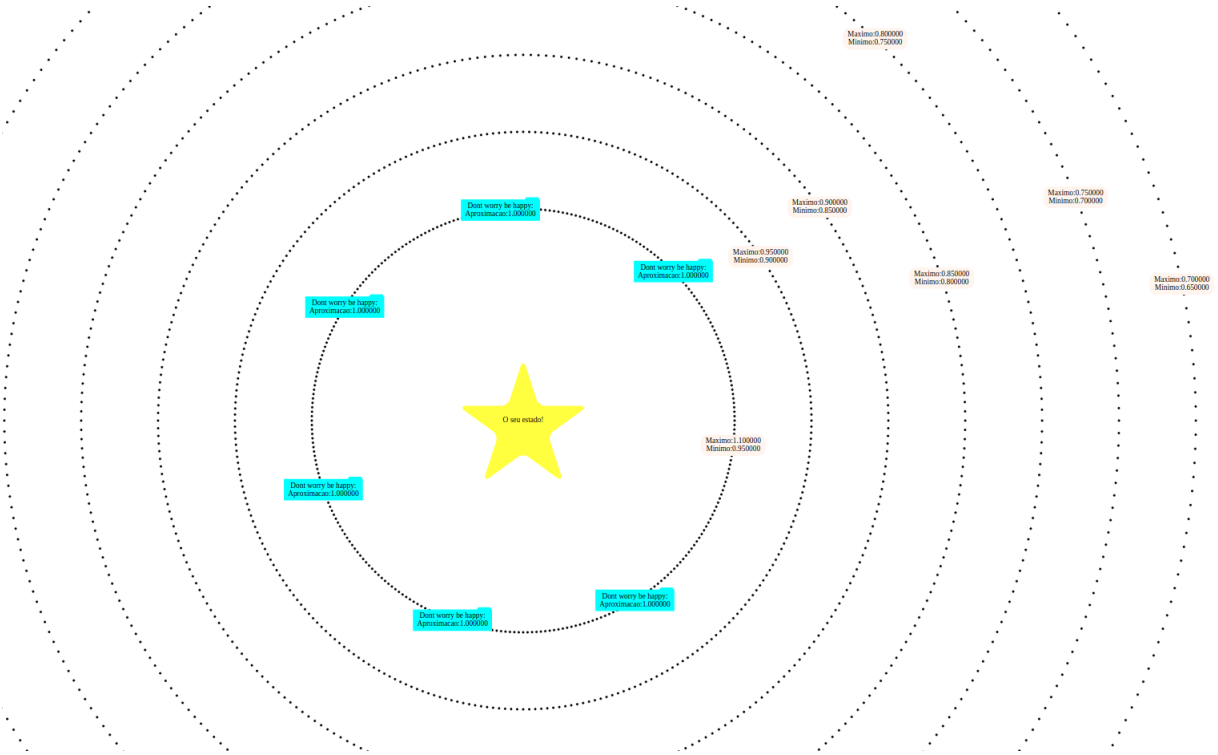


Figura 4.6: Grafo Gerado Pedido 2

Informações da Música

Nome:	Dont worry be happy
Autores:	Bob Marley
Ano Lançamento:	1988
Editora:	Linda Goldstein
Letra:	Clique Aqui!
Estilo:	RandB
Letra:	Play!
Percentagem Acerto:	1.000000

Figura 4.7: Informação da Musica

Capítulo 5

Conclusão

Após a realização do trabalho e de ver todos os resultados obtidos, podemos afirmar que o trabalho foi concluído com sucesso. Todos os objetivos inicialmente traçados foram corretamente realizados. A nossa gramática foi construída segundo as restrições definidas e ainda tivemos em atenção a análise do problema para verificar os grupos existentes na gramática, isto é, tentamos encontrar os casos gerais, para que a nossa gramática não fosse demasiada especifica, o que não seria adequado.

Toda a ligação à base de dados, que foi feita através do motor MySQL como referido anteriormente, apresentou resultados satisfatórios, sendo que esta parte do trabalho apresentou um pouco mais de trabalho e de dificuldades, isto porque não estávamos habituados a lidar com a ligação entre C e o MySQL. Obrigou-nos a um estudo mais aprofundado para saber quais as ferramentas que tínhamos disponível para efetuar a tal ligação, e ainda, à realização de inúmeras estruturas em C, o que por si só, já apresentou alguma carga de trabalho. Mas mais uma vez, foi um objetivo concluído.

Quanto à apresentação de resultados efetuado através de grafos foi bem-sucedida, acrescentado até algum extra, como o clicar sobre um dos nodos que representava a música e aparecer toda a informação relativa à música, até com ligações para o utilizador visualizar a letra da música ou até poder ouvir a mesma. Mais uma vez foi uma experiência nova, o uso da API *graphviz*, o que requereu algum tempo de aprendizagem, mas ficamos elucidados com o potencial desta ferramenta e todos os diversos tipos de grafos que tínhamos a dispor bem como as diferentes ações que poderíamos arranjar para cada um.

Em suma, foi um trabalho que se mostrou interessante para o grupo, que nos permitiu aprofundar o nosso conhecimento nas mais diversas ferramentas, onde obtivemos os resultados pretendidos.