Relatório do Projeto em JAVA

Pedro Gonçalves (a82313) Rodolfo Silva (a81716)

Junho 2018

Universidade do Minho
Laboratórios de Informática 3
Grupo 47

Conteúdo

1 Introdução

Este relatório aborda a implementação do projeto realizado na primeira parte da disciplina, desta vez na linguagem JAVA. O projeto consiste em construir um sistema que permita analisar dados presentes em backups do Stack Overflow, e extrair informação útil desses mesmos dados como, por exemplo, o número de publicações num dado período de tempo, o número de novos utilizadores, entre outros.

Para isto, foram criadas estruturas de dados para armazenar a informação desses backups, carregados os dados para essas estruturas sendo depois possível responder às queries pedidas no enunciado.

2 Desenho da Solução

Para esta segunda parte do trabalho, e para responder corretamente às queries pretendidas, o grupo optou por uma implementação simples mas eficiente. Dividimos o trabalho por módulos, ou seja, separamos o código por pequenos pedaços reutilizáveis de tal forma que seja fácil tanto reaproveitar estes pedaços de código, como fazer a manutenção sem que tenha impacto direto nos outros. Deste modo, a programação torna-se mais inteligente e menos suscetível a erros. Na nossa implementação temos 6 classes fundamentais, sendo elas a Parser, que realiza o parse de todos os ficheiros, fornecidos pela equipa docente, para a nossa estrutura de dados, a Users_str, que guarda toda a informação necessária relativamente a um utilizador, a UsrXpost, com os dados sobre todos os posts em que um dado utilizador participa, a Posts_str, com toda a informação sobre um dado post, a PostXusr com toda a informação sobre os utilizadores que participam num post e a classe TAD_community. A classe TAD_community define as estruturas onde vão ser guardadas todos os dados que queremos sobre as publicações e utilizadores. Nesta, utilizámos cinco HashMap, sendo quatro desses para armazenar as classes em cima descritas e o restante para armazenar dados sobre o Tags.xml. Optámos por esta decisão, devido às condições que um HashMap nos oferece. As chaves desses HashMap's são o id da publicaçãoutilizadortag dependendo da HashMap a que pretendemos aceder.

Em relação ao parser, o grupo decidiu utilizar *SAX* pois consideramos como o método mais eficaz para as necessidades do projeto, sendo um parse rápido e eficiente em termos de uso de memória.

3 Classes

Nesta secção vamos explicar as principais classes do projeto, e as principais estruturas das mesmas.

3.1 Users_str

```
private long usrID; //ID do user
private long reputation; // Reputaçao desse user
private String bio; // Biografia do User
private String usrName; // User Name
```

Para cada utilizador é necessário guardar o id do utilizador, a sua reputação, a sua biografia e também o seu nome de utilizador.

3.2 UsrXPost

```
private long post_total;
private ArrayList<help_struct_1> postID;
private ArrayList<help_struct_1> questionID;
private ArrayList<String> tags;
```

Para cada utilizador é necessário guardar a informação de todas as publicações em que um utilizador participa, como o número de publicações, todos os id's das publicações em que participa, todos os id's das perguntas em que participam e as tags das publicações em que os utilizadores participam.

3.3 Posts_str

```
private long postID; //ID do Post
private long pUsrID; // ID do user que fez o post
private long pType; //Tipo de Post *1 = Pergunta || *2 = Resposta
private long parentID; // Caso seja uma resposta, vai conter o ID da pergunta a qual esta a responder
private long score; // Score do post
private long comments; // Numero total de comentarios do post
private String titulo; // Caso seja uma Pergunta, vai conter o seu Titulo
private String tags; // Caso seja uma Pergunta, vai conter as tags associadas a ela
private long dataNumber; // Data de publicaçao concatenada, i.e., 2010-10-30 -> 20101030
```

Para cada publicação é necessário guardar o id da publicação, o id do utilizador que criou a publicação, o tipo de publicação, o id da publicação mãe, a pontuação da publicação, o número de comentários, o título as tags e também a data de publicação.

3.4 PostXUsr

```
private long dataPergunta;
private long numRespostas;
private ArrayList<help_struct_2> respID;
```

Para cada publicação é necessário guardar os id's dos utilizadores que participam numa dada pergunta.

3.5 Help_Struct_1

```
public class help_struct_1 implements Comparable<help_struct_1>
{
    private long id;
    private long data;
```

Estrutura que vai ser usada nas ArrayList's da estrutura UsrXPost, esta é usada para se poder guardar informação sobre os diferentes posts e questões na qual um user participou, neste caso esta-se a guardar o identificador do post e a data para facilitar as respostas as queries

3.6 Help_Struct_2

```
public class help_struct_2 implements Comparable<help_struct_2>
{
    private long idResposta;
    private long idUser;
    private long data;
```

Estrutura que vai ser usada na ArrayList da estrutura PostXUsr, esta é usada para se guardar a informação correspondente a uma resposta da pergunta a que esta associada, tal como o seu identificador, o identificador do utilizador que a publicou e a data de publicação, que vem depois facilitar a resposta a certas queries

4 Conclusão

Em suma, concluimos o projeto com sucesso. Através do parse dos ficheiros xml conseguimos extrair toda a informação relevante para o projeto, passando esta para uma estrutura de dados pensada com o obletivo de reponder às querys com a maior eficiência. Tínhamos como objetivo responder de maneira precisa e eficaz a todas as querys e também melhorar a eficiência relativamente ao projeto em C, e consoante os resultados pensamos que atingimos esses objectivos.