##### INF 2920 – Computação em Nuvem

## Trabalho 2 – WordCount

**Objetivo**

O trabalho consiste na implementação de um exemplo de utilização da arquitetura MapReduce. Este trabalho tem por objetivo contar o número de vezes que as palavras presentes nas obras de Shakespeare se repetem. Além de contabilizar as palavras, também será analisado o tempo de execução em diferentes números de instâncias.

**Tecnologias Utilizadas**

- Amazon Elastic MapReduce

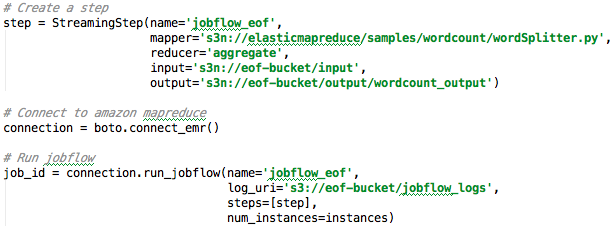
- Python 2.7.2

- boto - [Python interface to Amazon Web Services](http://code.google.com/p/boto/)

**Procedimento**

A primeira ação do trabalho foi a criação do bucket no *Amazon S3*, com o nome de *eof-bucket* (Eduardo de Oliveira Ferreira - bucket) e as pastas *input*, *mapper* e *jobflow\_logs*. A pasta *input* é responsável por armazenar todos os arquivos de texto a serem analisados pelo contador de palavras, no caso as obras de Shakespeare. A pasta *mapper* é responsável pelo arquivo de mapeamento das palavras. A pasta *jobflow\_logs* é necessária pois a *api* *boto* necessita de um diretorio para adicionar os *logs* gerados durante o *job.* Durante a execução do programa, ocorre a criação da pasta *output*, com os dados resultantes da contagem das palavras.

O próximo passo foi montar um script em Python para realizar o *MapReduce*. A *Amazon* disponibiliza aos desenvolvedores Python uma *api* chamada de *boto*. O *boto* permite ao usuário utilizar os serviços disponíveis pela *AWS*. No caso deste trabalho, iremos somente utilizar o serviço de *MapReduce*. Na Figura 1 está parte da codificação do *script* em Python que utiliza a *api boto*.

Figura 1 – Script em Python utilizando *boto*

Na Figura 1 podemos observar que o primeiro elemento a ser inicializado é um objeto do tipo *StreamingStep.* Esta classe esta presente em *boto.emr.step.StreamingStep* e é responsável por especificar um *step* do tipo streaming. Os parâmetros que devem ser passados para a inicialização do objeto são: *name*, *mapper*, *reduce*, *input* e *output*. O parâmetro *name* serve pra especificar o nome do *step* a ser criado. O parâmetro *mapper* é especificado a *URI* do diretório do arquivo de mapeamento no *bucket*. O *reducer* é responsável por pegar a saída do *mapper* e organiza-la de alguma forma especificada pelo usuário. Na figura é passado *aggregate*, que soma o número de vezes que as palavras aparecem no arquivo de saída do *mapper*. O parâmetro *input* recebe a URI dos arquivos a terem suas palavras contadas. Por último, o parâmetro *output* recebe a URI pra onde o resultado da contagem será colocado ao terminar.

Ao definir os campos obrigatórios do objeto do tipo *StreamingStep*, deve-se conectar ao serviço de *MapReduce* da *Amazon*. O método *boto.connect()* é responsável por isso. Após se conectar, deve-se utilizar o método *run\_jobflow* passando os parâmetros *name*, nome do *job*, *log\_uri*, URI do diretório onde os *logs* gerados devem ser colocados, *steps*, os *steps* que serão executados nete *job*, e *num\_instances*, o números de instâncias que irão executar o *job*. Definido os campos e executado o método *run\_jobflow*, é iniciado na *AWS* o *job* especificado. Este método tem como retorno o id do *job* criado.

**Analise dos resultados**

**Tecnologias Utilizadas**

- Amazon Elastic MapReduce

- Python 2.7.2

- boto 2.0 - [Python interface to Amazon Web Services](http://code.google.com/p/boto/)

**Testes**

O trabalho foi executado em três testes, cada um com um número diferente de instâncias. O número de instâncias utilizadas foram: 1, 10 e 20. Durante o desenvolvimento do trabalho, foi observado que entrado do programa (as obras de Shakespeare) era relativamente pequena para que se pudesse observar uma diferença nítida. Devido a isso, esses testes foram rodados com entradas sessenta vezes maiores que todas as obras de Shakespeare. Dessa forma pode-se observar os seguintes tempos de execuções:

* 1 instância = 11 minutos
* 10 instância = 6 minutos
* 20 instancia = 6 minutos

Podemos perceber que por aumentarmos o número de instâncias, tivemos um ganho significativo de tempo. Porém o tempo de execução dos Jobs com 10 e 20 instâncias foram semelhante. Acredito que a diferença de tempo pudesse ficar mais definida com entradas maiores que a utilizado. Pois, sem sombra de dúvidas, quanto maior for o número de instâncias utilizadas em um job, menor será seu tempo de execução.

**Resultados**

- As 10 palavras mais usadas foram: *the* (27843), *and* (26847), *i* (22538), *to* (19883), *of* (18307), *a* (14800), *you* (13928), *my* (12489), *that* (11563), e *in* (11183).

- Shakespeare escreveu mais sobre o amor do que ódio. Foram 2198 ocorrências de *love* contra 179 de *hate.*

- Em relação a comentar sobre vida ou morte, Shakespeare escreveu 900 a palavra *death* e 863 *life*.

- Em relação sobre vida ou morte, Shakespeare escreveu 8760 a palavra *not* e 7147 *be*.