Sistemas Distribuídos (noturno) – Q2/2019
Exercício Programático 3

1. Definição

Neste EP você desenvolverá um sistema que permita a um programa cliente requisitar, a uma arquitetura Map-Reduce, a criação de um índice invertido de links (semelhante a uma das atividades do PageRank do Google) seguindo o fluxo mencionado abaixo.

Fluxo:

1. O cliente envia uma requisição, contendo uma lista L de URLs, para um nó denominado coordenador. A lista será lida de um arquivo da pasta local do cliente.
2. O coordenador recebe a lista L e a divide em M listas L_1, L_2, …, L_l, …, L_M.
3. O coordenador envia cada lista L_l a um nó diferente, denominado *mapper*.
4. Cada *mapper*$_l$ recebe a parte L_l , realiza uma atividade (ver 1.1.) e envia o resultado da atividade a um nó denominado *reducer*.
5. O *reducer*, somente após receber o resultado de todos os M *mappers*, gera um índice invertido de links (ver 1.2.) e envia para o cliente.
6. O cliente recebe o índice invertido do *reducer* e o armazena em um arquivo na mesma pasta local do ponto 1. O arquivo poderá ser visualizado pelo professor.

1.1. A atividade do *mapper*

Deverá conter, no mínimo, as seguintes operações:

a) Receber em uma requisição uma lista de URLs do coordenador.
b) Baixar e armazenar na memória a página Web de cada URL da lista. Pode usar como exemplo o código em [1].
c) Obter todos os links que existem no conteúdo da página baixada. Para isso, pode usar uma livraria externa, como Jsoup (e.g., [2]) ou utilizar um parser com expressões regulares (e.g., [3]).
d) Criar uma estrutura que contenha, para cada URL da lista, todos os links extraídos dessa URL. Na seção 1.3. poderá observar um exemplo dessa estrutura.
e) Enviar a estrutura ao *reducer*.

1.2. A atividade do *reducer*

Deverá conter, no mínimo, as seguintes operações:

a) Receber a estrutura de cada *mapper*.
b) Verificar se já recebeu todas as estruturas dos M *mappers.*
c) Somente após receber as M estruturas, criar o índice invertido de links, como mostrado na Seção 1.3, e ordená-lo de forma decrescente pela quantidade.
d) Enviar o índice ordenado para o cliente.

* Assuma que o cliente conhece o endereço e porta do Coordenador.
* Assuma que o Coordenador conhece o endereço e porta de todos os Mappers.
* Assuma que cada Mapper conhece o endereço e porta do Reducer.
* Assuma que o Reducer **não** conhece diretamente o endereço e porta do Cliente.

A execução do programa deverá poder ser realizada com um número arbitrário de nós *mappers*. Na avaliação, será exigida a sua execução com os seguintes nós: 1 cliente, 1 coordenador, 3 (três) *mappers* e 1 *reducer*. Cada um desses nós deverá rodar em pelo menos 2 máquinas físicas, com IPs diferentes em 6 processos (JVM) diferentes. Não serão aceitos trabalhos que executam em somente uma máquina (mesmo sendo diferentes máquinas virtuais).

1.3. Criação do índice invertido de links

A estrutura criada na operação d) do *mapper* permite, dada uma URL X, conhecer as URLs que X aponta.

Por exemplo, vamos supor que o *mapper1* baixou e processou o www.site1.br, enviando ao reducer a seguinte estrutura:

www.site1.br: {www.site2.br, www.site3.br}

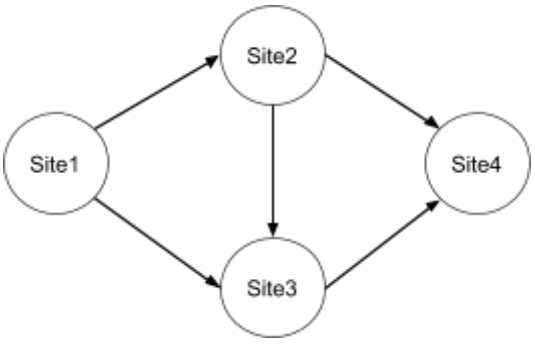
O *mapper2* baixou e processou o www.site2.br enviando ao reducer a seguinte estrutura:

www.site2.br: {www.site3.br, www.site4.br}

E o *mapper3* baixou e processou o www.site3.br, enviando ao reducer a seguinte estrutura:

www.site3.br: {www.site4.br}

Se quiser ver os apontamentos como um grafo, o exemplo acima define a figura abaixo.



Já o índice invertido criado na operação c) do *reducer* permite, dada uma URL Y, conhecer que URLs apontam para Y (exatamente o contrário da estrutura do mapper). Para o exemplo acima, o índice invertido de links (ordenado pela quantidade de apontamentos) será:

www.site3.br: {www.site1.br, www.site2.br} // tamanho 2
www.site4.br: {www.site2.br, www.site3.br} // tamanho 2
www.site2.br: {www.site1.br} // tamanho 1
www.site1.br: {} // tamanho 0

Note que o www.site4.br não foi baixado por nenhum mapper, porém apareceu na lista pelos link extraídos no item c).

2. Entrega

A entrega poderá ser realizada por um **grupo de até duas pessoas** e consistirá em um relatório e o código fonte do programa a ser entregue pelo TIDIA, na aba atividades.

O relatório deverá ter obrigatoriamente as seguintes seções:

a) Nome e RA dos participantes
b) Formato da mensagem encaminhada do cliente para o coordenador
c) Formato da mensagem encaminhada do coordenador para o mapper
d) Formato da mensagem encaminhada do mapper para o reducer
e) Formato da mensagem encaminhada do reducer para o cliente.
f) Explicação em “alto nível” de como o reducer espera os M mappers
g) Links dos lugares de onde baseou seu código (caso aplicável).

O código fonte (e na avaliação) deverá apresentar claramente na console o fluxo para cada nó. Por exemplo:

Console cliente: enviando lista L com 35 URLs
Console *coordenador*: recebendo Lista L com 35 URLs, enviando 12 mapper1, 12 mapper2, 11 mapper 3
Console mapper1: recebendo lista com 12 URLs … processando
Console mapper2: recebendo lista com 12 URLs … processando
Console mapper3: recebendo lista com 11 URLs … processando
Console mapper3: enviando estrutura ao reducer
Console reducer: recebendo estrutura do mapper3
Console mapper1: enviando estrutura ao reducer
Console reducer: recebendo estrutura do mapper1
Console mapper2: enviando estrutura ao reducer
Console reducer: recebendo estrutura do mapper2
Console reducer: todas as estruturas recebidas, criando índice invertido e ordenando
Console reducer: enviando índice invertido ao cliente
Console cliente: índice invertido recebido, armazenado na pasta X.

3. Datas

A entrega tanto do relatório quanto do código fonte deverá ser realizada até o dia 22 de agosto às 19.00 (antes da aula prática) **somente via TIDIA**. Envios por email ou outra forma não serão aceitos.

4. Avaliação

A avaliação será realizada na aula prática do dia 22 de agosto. De forma aleatória o professor escolherá os participantes a serem avaliados e lhes fará perguntas. No final, a nota individual será a do grupo. Caso seu grupo não seja selecionado, a avaliação será realizada em um dia a ser agendado, com o código entregue no dia 22 de agosto.

No dia da avaliação, deixe pronto o sistema para funcionar e o código fonte aberto para responder as perguntas.

Observações:

- Caso o participante não esteja presente na hora da chamada terá nota zero.
- Caso o código não compile ou não esteja pronto para mostrar, terá nota zero, independentemente de ter entregue o relatório.

5. Atrasos

Não haverá correção para entregas com atraso. A nota será zero.

7. Bônus

O participante terá um bônus de 2 pontos se os 6 nós (cliente, coordenador, 3 mappers e reducer) forem executados em 6 instâncias Ubuntu (do EC2 da Amazon) diferentes. Caberá ao participante mostrar o funcionamento das instâncias na hora da apresentação.

Para ter acesso aos benefícios da Amazon (para alunos da UFABC) realize o cadastro com seu email da UFABC no site: https://aws.amazon.com/pt/education/awseducate/

8. Links recomendados

[1] https://docs.oracle.com/javase/tutorial/networking/urls/readingURL.html
[2] https://jsoup.org/cookbook/extracting-data/example-list-links
[3]https://www.mkyong.com/regular-expressions/how-to-extract-html-links-with-regular-expression/

9. Ética

Cola, fraude, ou plágio implicará na nota zero a todos os envolvidos em todas as avaliações e exercícios programáticos da disciplina.