UNIVERSIDADE DE BRASÍLIA
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO
CURSO DE CIÊNCIA DA COMPUTAÇÃO

PAULO DA CUNHA PASSOS
PEDRO HENRIQUE SOUZA TOLENTINO

IMPLEMENTAÇÃO DE PROTOCOLO
Distributed Hash Table (DHT)

BRASÍLIA-DF
2015

PAULO DA CUNHA PASSOS
PEDRO HENRIQUE SOUZA TOLENTINO

IMPLEMENTAÇÃO DE PROTOCOLO
Distributed Hash Table (DHT)

Trabalho apresentado a disciplina de Teleinformática e Redes 1 como requisito parcial para aprovação na disciplina.

BRASÍLIA-DF
2015

INTRODUÇÃO

A arquitetura de redes chamada Peer-to-peer(P2P), ou também conhecida como Ponto-a-ponto é uma arquitetura em que cada um dos pontos da rede funciona ora como cliente ora como servidor, permitindo compartilhamentos de serviços e de dados sem a necessidade de um servidor central. Nessa arquitetura cada computador da rede é um nó, e é responsável por disponibilizar uma parcela dos recursos da rede ao mesmo tempo que usa recursos disponibilizados por outros nós da rede. O que dife da arquitetura predominante nas redes, a arquitetura cliente-servidor, onde apenas o servidor fornece o conteúdo/serviço a rede e os clientes apenas consomem (KUROSE; ROSS, 2003).

Um dos principais problemas em redes P2P é a localização de um nó que armazena um determinado dado. Como os nós entram e saem do sistema constantemente, os dados podem migrar de um nó para outro, de forma a dificultar a localização de tais dados (MADRUGA et al, 206). Portanto, para um funcionamento eficiente da rede é preciso garantir que todos os dados inseridos possam ser encontrados.

Uma das propostas para resolver esse problema é o uso de uma técnica chamada *Distributed Hash Table* (DHT), que possibilita a indexação e a localização eficiente de dados armazenados, desde que um identificador seja relacionado a este dado (ROCHA; ROSS, 2005).

As DHT's, como são conhecidas as redes que usam a técnica de *Distributed Hash Table,* disponibilizam uma função de consulta similar à encontrada em tabelas de *hash* tradicionais, sendo que as chaves são distribuídas pelos diversos nós da rede. Tal função realiza basicamente uma pesquisa por uma determinada chave e retorna o objeto associado a esta chave. Os objetos podem ser arquivos, serviços, dados, ou qualquer outro valor indexado a chave(ROCHA; DE BONA, 2009). Nessas redes as chaves ou os identificadores de cada nó segue um determinado critério distribuição de forma que cada um seja único(a). E então, cada nó mantém uma lista contendo um

subconjunto de nós, conhecidos nesta rede, e seus identificadores. Isso se faz necessário para facilitar o rastreamento de outros nós, sem que seja necessário que cada nó tenha que rastrear todos os outros nós da rede.

Uma forma de facilitar o rastreamento na rede é organizar os pares de nós em uma forma abstrata de circulo, e usar o DHT circular. Essa abordagem é um caso especial de uso de redes sobrepostas, que é a formação de uma rede lógica abstrata de nós acima de uma rede física.

Usando redes sobreposta pode-se ter cada nó responsável por rastrear apenas seu sucessor, devendo estar ciente apenas de dois pares de nós na rede, seu sucessor imediato e seu antecessor imediato. Isso diminui muito a quantidade de informação que cada nó deve gerenciar, porém, nesse caso, para um nó encontrar outro nó responsável por uma determinada chave ele deve em média mandar X/2 mensagens, em um rede DHT que contenha X nós. O que ainda pode ser uma quantidade muito grande de mensagens, dependendo do tamanho da rede em que esse nó está inserido. Portanto, pode-se realizar um aprimoramento dessa técnica de uso de redes sobreposta circular, acrescentando atalhos de forma que cada nó rastreei não só seu sucessor imediato, mas um número relativamente peque de outros nós no circulo abstrato. E esses atalhos serão usados para rotear as mensagens enviadas até se chegar ao nó mais próximo da chave que se busca. Essa solução permite reduzir significativamente o número de mensagens expedidas para uma dada solicitação (KUROSE; ROSS, 2003).

DESENVOLVIMENTO

A aplicação foi desenvolvida em *python*, e é composta pelos seguintes arquivos:

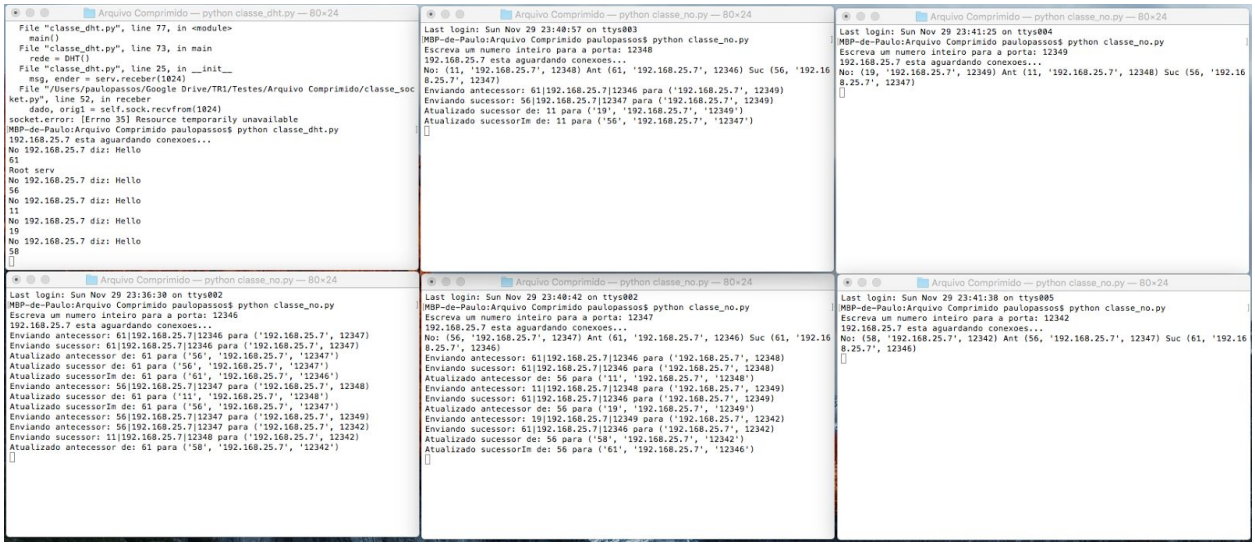
- classe_dht.py - Classe responsável pela criação da DHT e que realiza o papel do servidor *rendezvous*
- classe_no.py - Classe que implementa um nó e as respectivas funções de acesso à rede.

- classe_socket.py - Classe que contém as funções de tratamento dos sockets UDP. Nela foram implementadas as funções enviar e receber, responsáveis por pelo envio e recebimento de mensagens via socket UDP com a devida verificação se o envio e o recebimento foram bem sucedidos.

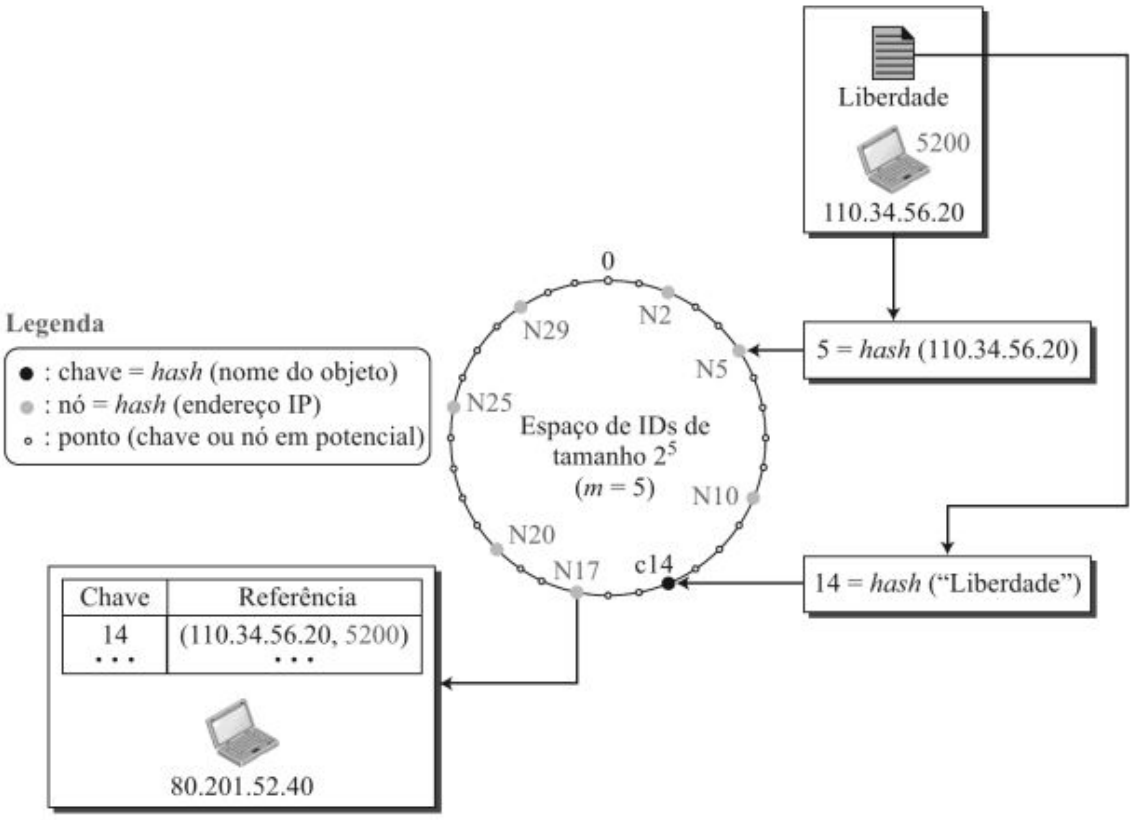
A ordem de execução consiste na inicialização do servidor *rendezvous* (execução do arquivo classe_dht.py). Uma vez iniciado, este fica ao aguardo nós que queiram se conectar a ele. Em seguida, realiza-se a inserção do nó *root* (primeira execução do arquivo classe_no.py). Uma inicializado o primeiro nó e este consegue realizar a conexão com o *rendezvous*, este é então declarado como nó *root* da rede.

Conforme demais nós são inseridos na rede, eles se comunicam com o *root*, de forma que ele que realiza a inserção do novo nó na estrutura do DHT.

Cada nó inserido na rede tem um nome de arquivo o qual referencia. E uma vez inseridos na DHT estes podem realizar busca de nomes de outros arquivos que estejam inseridos em outros nós na rede. A busca se dá percorrendo nó por nó na rede, a partir daquele que realizou a requisição, e verificando se o nome buscado corresponde ao existente no nó avaliado. Caso a busca encontre resultado o endereço do nó que contém o arquivo é exibido.



O algoritmo de construção do DHT é baseado nos algoritmos de inserção de elementos em uma lista duplamente encadeada e fechada. Assim, como o algoritmo para percorrer a rede e realizar pesquisa na rede é baseado nos similares relacionados a lista encadeadas.



CONCLUSÃO

O objetivo do grupo era construir um DHT circular similar ao DHT Chord com uma tabela de atalhos para diminuir o número de requisições na rede, evitando inundações. Porém, encontrou-se dificuldade em construir a função responsável por construir a Finger Table e não foi possível sua elaboração antes da data de entrega do presente trabalho. Também não foi possível, fazer a verificação de estabilidade da rede por meio de pings. Tentou-se implementar essa função por meio de Threads, porém, a

pouca experiencia em programação concorrente e em linguagem Python, levaram a um grande desperdício de tempo sem a efetiva implementação de tais funções.

Portanto, não foi possível ao grupo atender todas as especificações do trabalho no tempo desejado. Sendo apresentado até a presente data a implementação parcial.

REFERÊNCIAS BIBLIOGRÁFICAS

KUROSE, James F.; ROSS, Keith W. **Redes de Computadores ea Internet: Uma nova abordagem**. Addison Wesley, 2003.

MADRUGA, Marcos; BATISTA, Thais V.; GUEDES, Luiz A. Uma Arquitetura P2P Baseada na Hierarquia do Endereçamento IP. In: **Simpósio Brasileiro de Redes de Computadores**. 2006.

ROCHA, Vladimir; ROSS, Eric. Uma Estrutura Escalável e Eficiente para Buscas por Intervalo sobre DHTs nas Redes P2P. In: **I Workshop de Redes Peer-to-Peer-WP2P**. 2005.

ROCHA, Pedro Eugênio; DE BONA, Luiz Carlos Erpen. Um Sistema de Arquivos Compartilhado em Nıvel de Usuário Baseado em Tabelas Hash Distribuıdas. 2009