Victor Meira Pinto 05/04/19

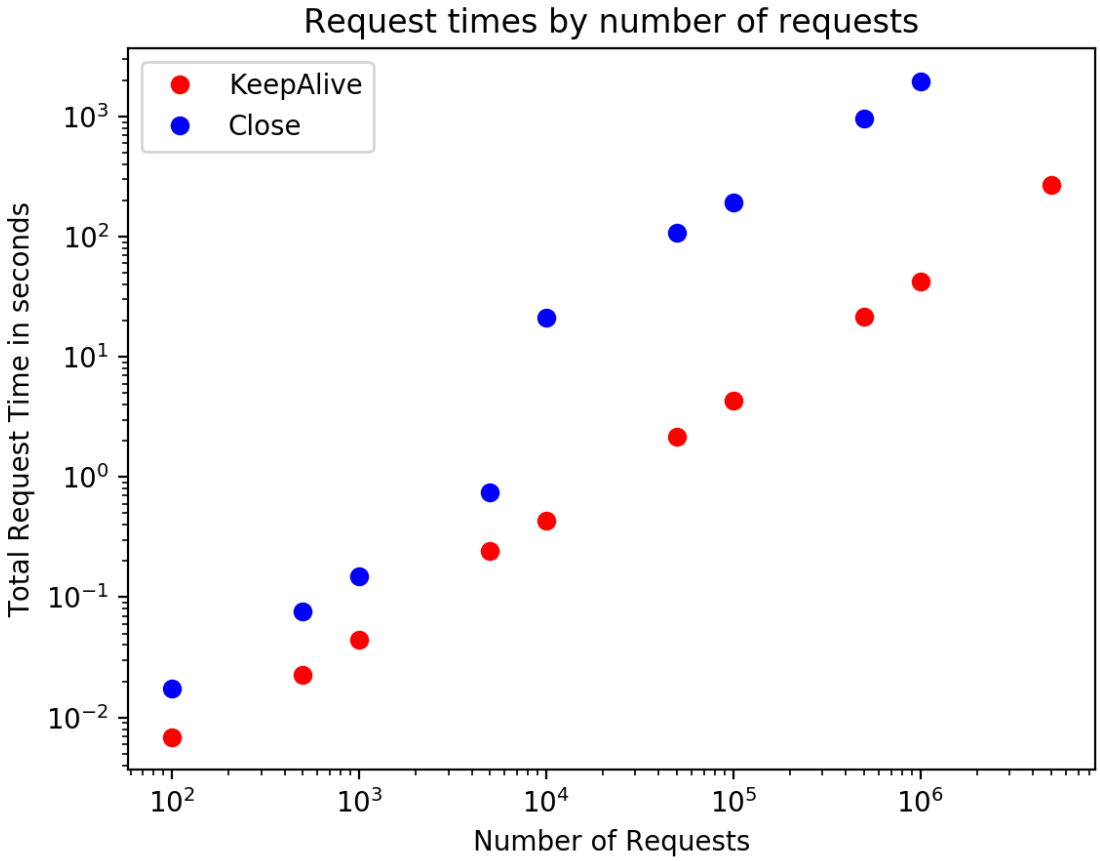
Trabalho 1 - Cliente Servidor em Lua Sockets

Implementando os dois tipos diferente de interação entre o cliente servidor, uma onde a conexão é fechada após cada request, que chamarei de *close*, e outra onde a conexão é mantida aberta até que o cliente feche a mesma, que chamarei de *keep alive.* Devido ao tempo que se demora para fechar e abrir uma nova conexão, o esperado é que a política de *close* tenha tempos de respostas maiores do que a de *keep alive,* na média. Para isso, foi desenvolvido em Lua, utilizando Lua Sockets, um cliente e um servidor que pudessem receber e enviar requests nas duas políticas, e que com isso o tempo pudesse ser medido para confirmar a teoria de que *close* irá performar pior que a *keep alive*.

No lado do servidor, o script em Lua recebe um parâmetro de policy que descreve qual política deve ser utilizada, e com isso, já abre uma conexão e fica esperando o cliente se conectar ao socket. Ao receber uma requisição, ele manda de volta uma string de 1K para o cliente, e dependendo da política, fecha a conexão e espera uma nova, ou espera uma nova requisição do mesmo cliente. Como não há uma forma de saber se a conexão está fechada, o servidor ao não receber nada na chamada receive, também fecha a conexão, já que ao não receber nenhuma requisição, o cliente não está mais conectado.

Do lado do cliente, o script em Lua também recebe o mesmo parâmetro de policy, e também recebe o número de requisições que devem ser feitas, assim como, onde ele deverá escrever o tempo final das consultas. O cliente abre a conexão com o servidor, e manda uma requisição com qual string deve receber de volta. Após recebimento, dependendo da configuração de política, o cliente fecha a requisição e abre uma nova, ou mantém a mesma.

Rodando para uma quantidade de números de requisições diferentes, obteve-se o gráfico abaixo, gerado a partir de um script em python lendo os arquivos gerados, utilizando a biblioteca matplotlib. O gráfico está com escala logarítmica para que seja mais fácil visualizar as diferenças. Não foi gerada para *close* o teste com 5,000,000 devido ao tempo elevado do teste.



Nota-se claramente que o tempo para as requisições para Keep Alive mantém-se sempre abaixo do close, mais ou menos por um fator de 5 vezes menor em tempo total. Isso se dá pelo fato de que o fechamento e abertura de uma nova conexão aumenta consideravelmente o tempo de resposta devido a forma que o protocol TCP é implementado via comunicação entre sockets. Logo, pode-se concluir que para sistemas distribuídos e de alta escala, deve-se sempre preservar a conexão com o cliente para que haja um tempo de resposta sempre o menor possível.