UNIVERSIDADE FEDERAL
DE ALAGOAS

UNIVERSIDADE FEDERAL DE ALAGOAS
CURSO DE CIÊNCIA DA COMPUTAÇÃO

CAIO MASCARENHAS SOARES
FILIPE MARANHÃO RAPOSO RODRIGUES
GABRIELA BATISTA TENÓRIO DAS NEVES
LAIS SOUZA DE ALMEIDA

REDES DE COMPUTADORES
PROFESSOR ALMIR PEREIRA GUIMARÃES

MACEIÓ - ALAGOAS
2023

PROJETO REDES DE COMPUTADORES
APLICAÇÃO COM USO DE SOCKETS

Trabalho apresentado para a Disciplina Redes de computadores, pelo Curso de Ciência da Computação da Universidade Federal de Alagoas (UFAL), ministrada pelo Prof. ALMIR PEREIRA GUIMARÃES

Maceió 2023

SUMÁRIO

INTRODUÇÃO

A seguir será apresentado um relatório abordando as principais funcionalidades da aplicação, construída em python, de um sistema de rede com a implementação de SOCKET e THREAD. O sistema se utiliza de uma conexão Servidor-Cliente do tipo TCP/IP usando IPV4, e tem como propósito fazer duas máquinas interagirem entre si trocando informações.

Desenvolvimento

1 - Funcionalidades

Primeiramente importamos duas bibliotecas, a “socket” e a “threading”. A biblioteca **Sockets** serve para gerenciar os sockets de comunicação entre as duas máquinas, o que vai permitir a troca de informações cliente-servidor, usaremos o socket para fornecer informações como Porta, IP do host, entre outros, Tudo isso se faz necessário para estabelecer a conexão.

Por outro lado, a biblioteca **threading**, será responsável por gerenciar os trechos de código que estão sendo executados e enviados para o servidor, mantendo a execução eficiente e dinâmica para estabelecer a conexão feita pelos sockets.

2 - O que poderia ser implementado

Depois que foi estabelecida a conexão entre cliente e servidor podemos executar incontáveis algoritmos que dependem de 2 lados para serem executados, como jogos competitivos, quizzes, chats de conversa em tempo real, videoconferência…

No caso de um quiz, por exemplo, logicamente seria necessário um trabalho árduo no que se refere a algoritmos, consumo de API's e outras coisas mais, porém, em via de regra a conexão estabelecida via sockets e threading não mudaria drasticamente.

Resumidamente, depois que foi estabelecida uma conexão entre sockets, seria possível implementar diferentes algoritmos para trocarem informações, sem mudar necessariamente a base de conexão que já foi construída.

3 - Dificuldades encontradas

Ao longo do desenvolvimento deste projeto, enfrentamos algumas dificuldades que merecem destaque. Primeiramente, a configuração e a compreensão inicial das bibliotecas de sockets e threading em Python foram desafiadoras. Entender como criar e gerenciar sockets para estabelecer a comunicação entre as duas máquinas e implementar threading para manter a execução eficiente foi um processo de aprendizado significativo.

Ao implementar a função matemática para controlar o crescimento da barra de progresso, enfrentamos desafios na escolha e ajuste da função quadrática apropriada. Determinar os coeficientes corretos e compreender seu comportamento exigiu conhecimentos de álgebra e cálculo.

Em resumo, este projeto envolveu uma série de desafios que abrangeram desde a configuração de sockets e threading até a implementação de lógica matemática e componentes de interface gráfica. O processo de superar essas dificuldades contribuiu para um aprendizado significativo, visto que tais assuntos não tinham sido vistos até então no curso de ciência da computação, sendo portanto, de grande importância.

Código

- **Servidor (server.py)**

```python
def host_game(self, host, port):
    server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    server.bind(("0.0.0.0", port))
    server.listen(1)

    client, addr = server.accept()
    threading.Thread(target=self.handle_connection, args=(client,)).start()
    server.close()
```

```python
def connect_to_game(self, host, port):
    client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    client.connect((host, port))

    self.you = 'O'
    self.opponent = "X"
    threading.Thread(target=self.handle_connection, args=(client,)).start()
```

* **SOCKET.SOCKET()** essa instância do objeto socket, cria um Socket com determinadas características, que pode-se passar como parâmetro, no caso, precisaremos passar socket.AF_INET socket.SOCK_STREAM, indicam o endereço da rede será do tipo IPV4 e que vai usar TCP/IP protocol.
* **SOCKET.BIND()** essa instancia do objeto socket, associa estrutura socket e o endereço/porta do servidor
* **SOCKET.LISTEN()** essa instancia do objeto socket, o servidor entra no estado de aguardar por alguma solicitação de clientes que desejam se comunicar,recebe como parâmetro a quantidade de conexões que serão permitidas pelo protocolo TCP, esse número depende do tipo de aplicação e capacidade do servidor.
* **SOCKET.ACCEPT()** essa instancia do objeto socket, aceita o cliente que está tentando se conectar e retorna uma tupla, que indica o endereço do cliente que solicitou a conexão

***THREADING.THREAD()**Esse código cria uma nova thread para lidar com uma conexão de cliente, executando a função self.handle_connection(client) em paralelo.