分布式系统第一次作业

q2文档部分

1752854

赵羿昕

# 代码环境：

Python3.7

# 使用库：

import mmap

import os

import time

import BytesWrite

import socketserver

import socket

import threading

from mpl\_toolkits.mplot3d import axes3d #画3d图

import matplotlib.pyplot as plt #画图用

from multiprocessing import Process #main.py使用多进程开启服务器和客户端

# 核心函数：

## 服务器端

1. 服务器初始化：

def init():

    """

    初始化服务端

    """

    global g\_socket\_server

    g\_socket\_server = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)  # 创建 socket 对象

    g\_socket\_server.bind(ip\_port)

    g\_socket\_server.listen(5)  # 最大等待数

    print("服务端已启动，等待客户端连接...")

2.接收新连接：

def accept\_client():

while True:

        if len(g\_conn\_pool) < thread\_max:

            client, \_ = g\_socket\_server.accept()  # 阻塞，等待客户端连接

            # 加入连接池

            g\_conn\_pool.append(client)

            # 给每个客户端创建一个独立的线程进行管理

            thread = threading.Thread(target=message\_handle, args=(client,))

            thread.setDaemon(True)

            thread.start()

        else:

            time.sleep(0.1)

3.消息处理函数：

def message\_handle(client):

    """

    消息处理

    """

    global thread\_max

    global flag

    while True:

        bytes = client.recv(1024)                       #客户端消息

        if bytes.decode(encoding='utf8') == '5':

            thread\_max += 1

            rst = 'ok'

            print(thread\_max)

            client.sendall(rst.encode(encoding='utf8'))

        elif bytes.decode(encoding='utf8') == 'end':

            rst = 'end'

            client.sendall(rst.encode(encoding='utf8'))

            flag = 1

        else:

            msg = SearchWords(bytes)

            client.sendall(msg.encode(encoding='utf8'))

        if len(bytes) == 0:

            client.close()

            # 删除连接

            g\_conn\_pool.remove(client)

            break

## 客户端：

1.发信函数：

def searchWords(full = '0'):

    tcp\_client = socket(AF\_INET,SOCK\_STREAM)

    tcp\_client.connect(ip\_port)

    if full == '0':

        tcp\_client.sendall(msg.encode("utf-8"))

    else:

        message = full

        tcp\_client.sendall(message.encode("utf-8"))

    data = tcp\_client.recv(buffer\_size)

    #print(data.decode("utf-8"))   要看结果可以在这里

    tcp\_client.close()

2.绘图函数：

def draw(paint):

    i = 1

    x = []

    y = [[] for i in range(5)]

    while i < 6:

        j = 1

        x.append([i, i, i, i, i])

        while j < 6:

            y[i - 1].append(j)

            j += 1

        i += 1

    fig = plt.figure()

    ax = fig.gca(projection='3d')

    ax.set\_title("server/client time")

    ax.set\_xlabel("threads - server")

    ax.set\_ylabel("threads - client")

    ax.set\_zlabel("time")

    figure1 = ax.plot(x[0], y[0], paint[0], c='g')

    figure2 = ax.plot(x[1], y[1], paint[1], c='b')

    figure3 = ax.plot(x[2], y[2], paint[2], c='y')

    figure4 = ax.plot(x[3], y[3], paint[3], c='orange')

    figure5 = ax.plot(x[4], y[4], paint[4], c='r')

    plt.savefig(module\_path + "out\\result.png")

    plt.show()

3.客户端主函数：

def clientMain():

    n = 1

    count = 1

    paint = [[] for i in range(5)]

    while n < 6 and count < 6:

        li = []

        start = time.time()

        for i in range(n):

            li.append(Thread(target=searchWords))

        for i in li:

            i.setDaemon(True)

            i.start()

        for i in li:

            i.join()

        end = time.time()

        past = end - start

        paint[count-1].append(past)

        print(n)

        print(past)

        if n == 5 and count < 25:

            searchWords('5')

            n = 1

            count += 1

        elif n < 5 and count < 6:

            n += 1

        elif count >= 6:

            break

    searchWords('end')

    draw(paint)

# 执行结果（用自己的数据绘的图，根据新数据生成的图片在out文件夹下）：

