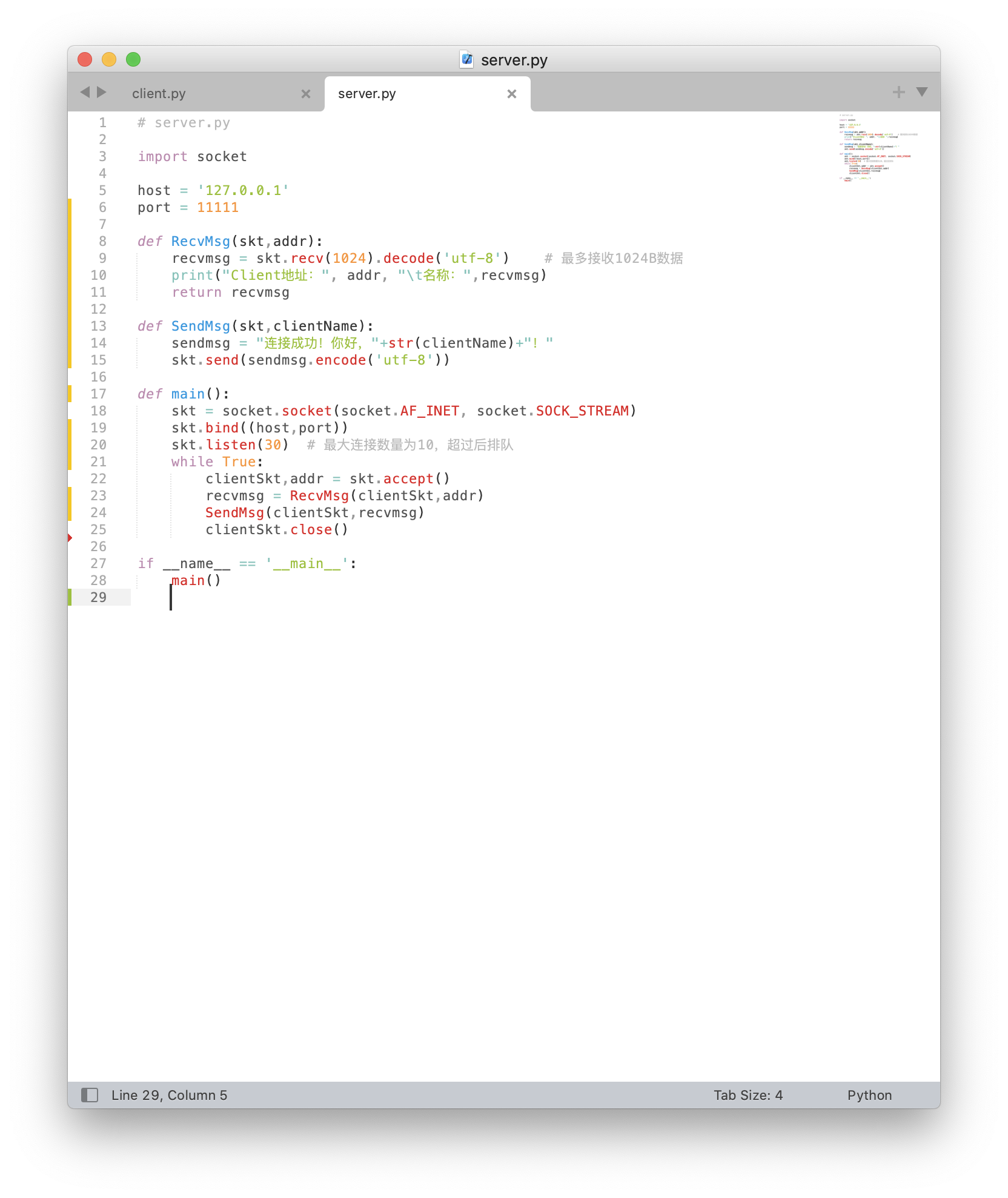
1. 源代码
2. 运行结果截图
3. 响应时间与并发请求量之间关系
4. 源代码
5. 服务器端



# server.py

import socket

host = '127.0.0.1'

port = 11111

def RecvMsg(skt,addr):

recvmsg = skt.recv(1024).decode('utf-8') # 最多接收1024B数据

print("Client地址：", addr, "\t名称：",recvmsg)

return recvmsg

def SendMsg(skt,clientName):

sendmsg = "连接成功！你好，"+str(clientName)+"！"

skt.send(sendmsg.encode('utf-8'))

def main():

skt = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

skt.bind((host,port))

skt.listen(30) # 最大连接数量为10，超过后排队

while True:

clientSkt,addr = skt.accept()

recvmsg = RecvMsg(clientSkt,addr)

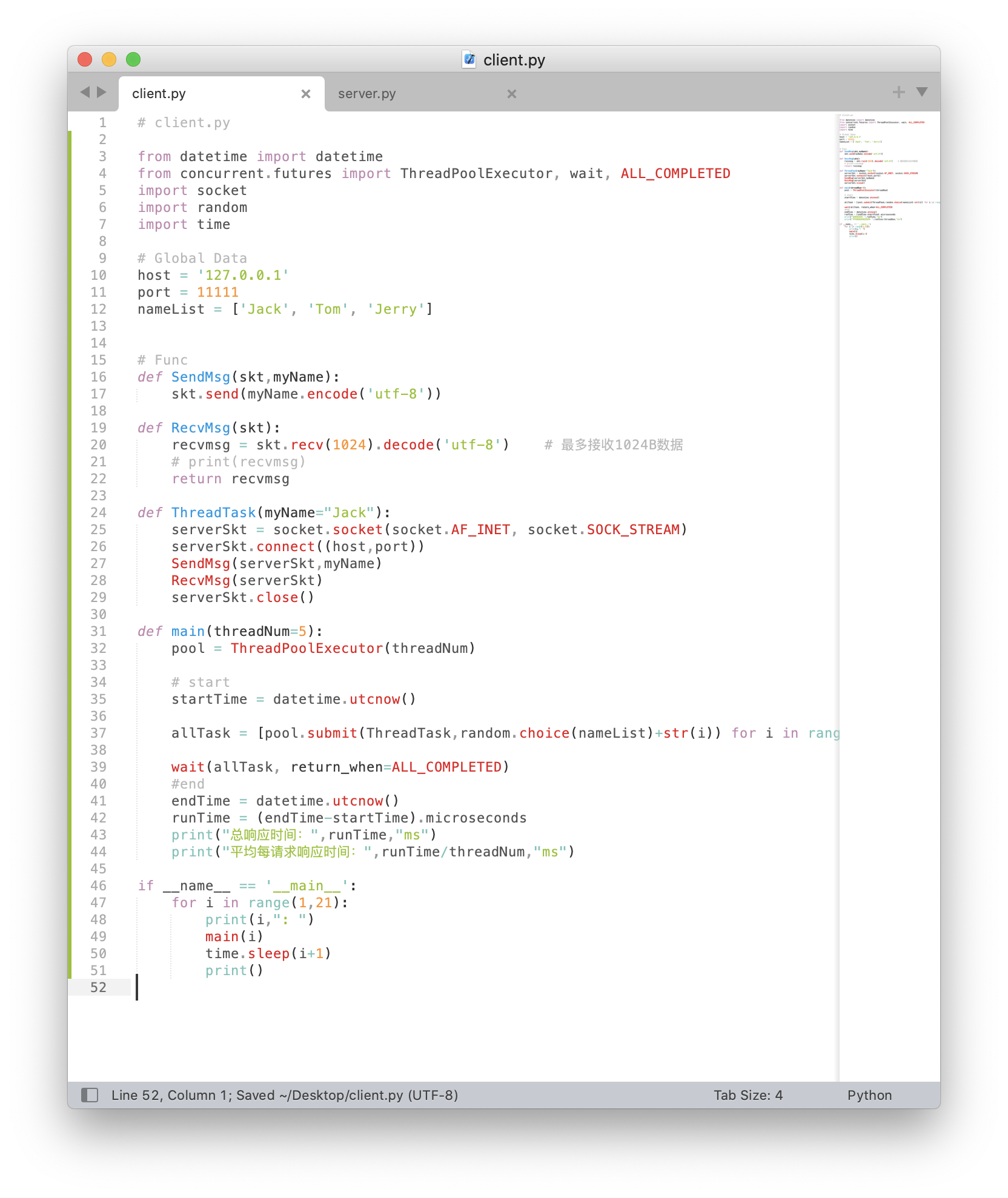
SendMsg(clientSkt,recvmsg)

clientSkt.close()

if \_\_name\_\_ == '\_\_main\_\_':

main()

1. 客户端



# client.py

from datetime import datetime

from concurrent.futures import ThreadPoolExecutor, wait, ALL\_COMPLETED

import socket

import random

import time

# Global Data

host = '127.0.0.1'

port = 11111

nameList = ['Jack', 'Tom', 'Jerry']

# Func

def SendMsg(skt,myName):

skt.send(myName.encode('utf-8'))

def RecvMsg(skt):

recvmsg = skt.recv(1024).decode('utf-8') # 最多接收1024B数据

# print(recvmsg)

return recvmsg

def ThreadTask(myName="Jack"):

serverSkt = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

serverSkt.connect((host,port))

SendMsg(serverSkt,myName)

RecvMsg(serverSkt)

serverSkt.close()

def main(threadNum=5):

pool = ThreadPoolExecutor(threadNum)

# start

startTime = datetime.utcnow()

allTask = [pool.submit(ThreadTask,random.choice(nameList)+str(i)) for i in range(threadNum)]

wait(allTask, return\_when=ALL\_COMPLETED)

#end

endTime = datetime.utcnow()

runTime = (endTime-startTime).microseconds

print("总响应时间：",runTime,"ms")

print("平均每请求响应时间：",runTime/threadNum,"ms")

if \_\_name\_\_ == '\_\_main\_\_':

for i in range(1,21):

print(i,": ")

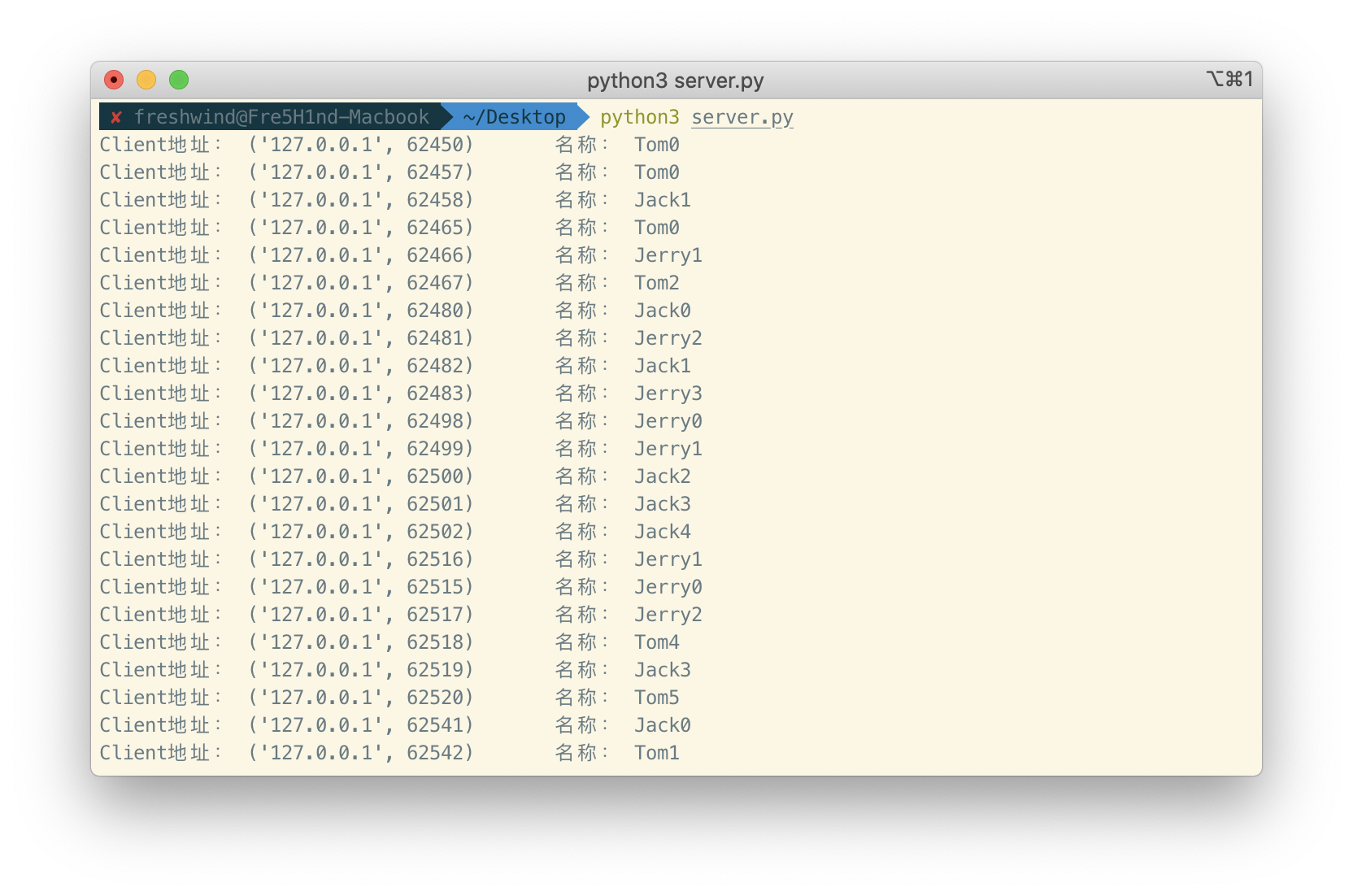
main(i)

time.sleep(i+1)

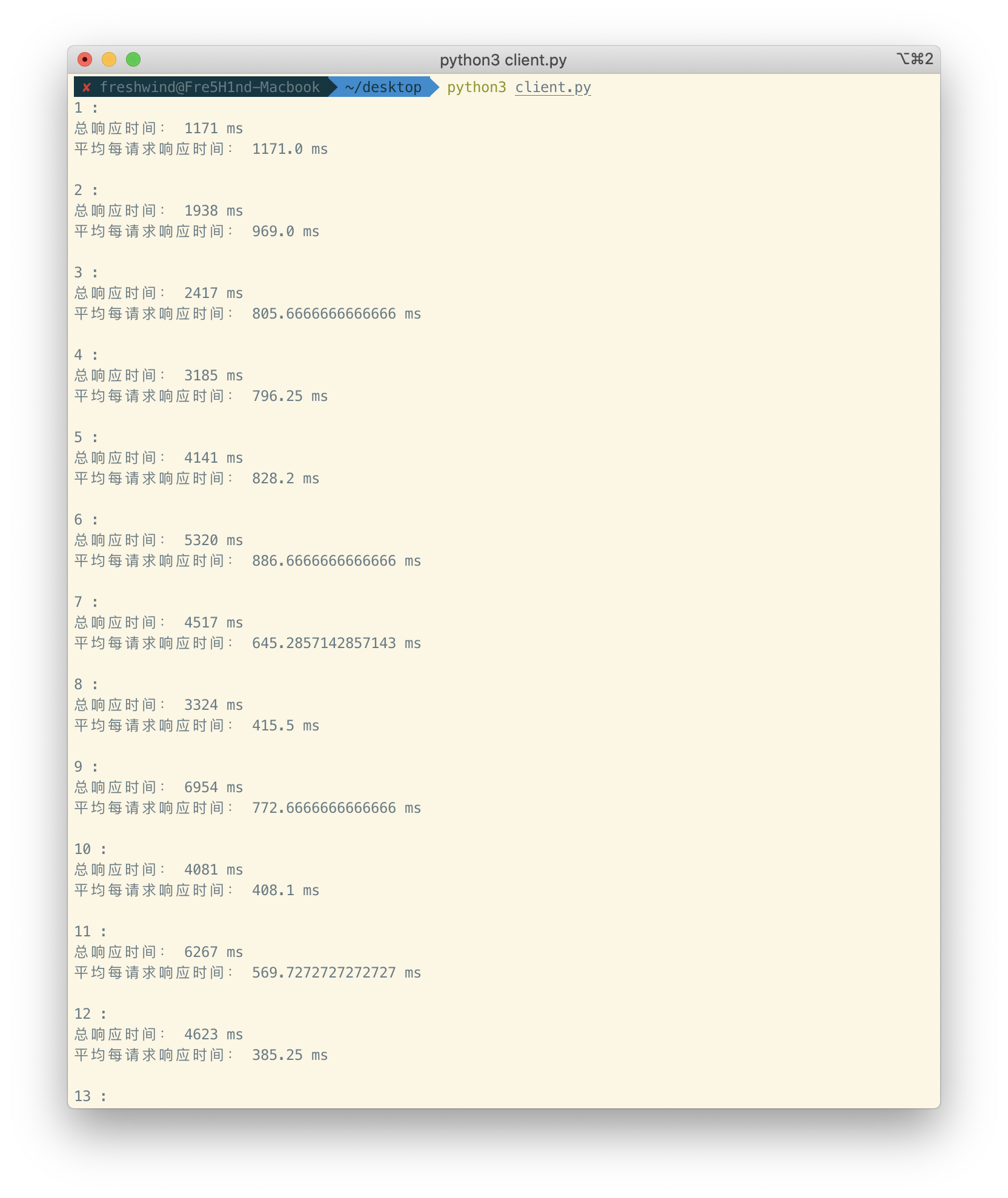
print()

1. 运行结果截图

* 服务器端



* 客户端



1. 响应时间与并发请求量之间关系

