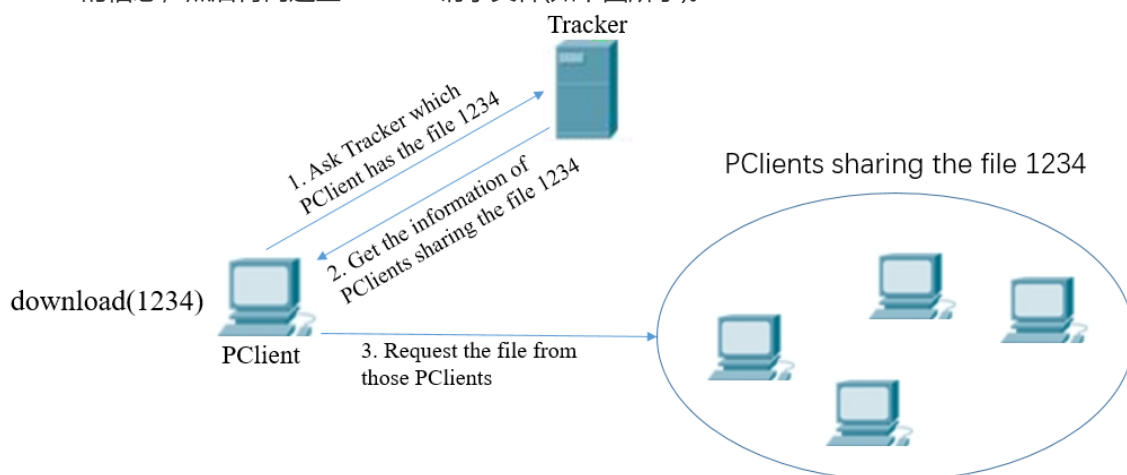


Project 2: P2P 文件传输系统

命题人: 张嘉兮, 杨睦圳

简介

P2P相较于传统的Server-Client模型, 可以更充分地利用不同节点的上传与下载带宽, 实现网络越大, 文件共享速度越快的目的。**在这个项目中, 你需要构建一个P2P文件传输系统, 按要求设计并实现其中Tracker与PClient的功能。**在这个系统中, PClient可以利用Tracker上记载的信息实现文件共享的目的; 当一个PClient想从P2P网络中下载某个文件时, 他需要先向Tracker请求拥有目标文件的其他PClient的信息, 然后再向这些PClients请求文件(如下图所示)。



除本文档中的限制与要求外, 请自行设计其中所有实现细节, 如一个文件如何进行拆分与传输后的组装、不同报文的报文头格式、PClient如何决定向哪些PClient请求文件等。

需要实现的内容

- Tracker.py (需实现整个Tracker功能)
 - Tracker用来存储PClient之间实现通信的信息, 如每个节点的地址、网络中正在共享的可被下载的文件、某个文件或文件块的持有者等
 - 你需要自行设计并实现PClient与Tracker之间的通信规则, Tracker上存储的具体信息(上面给出的例子仅供参考, 请根据自己的设计需求进行设计)以及提供给PClient的API
 - 我们在SimpleTracker.py中提供了一种非常简易的Tracker的设计, 若毫无头绪可尝试基于它进行改造
 - **注意: 在我们的设定中tracker不存储实际文件内容!!! 在我们的测试中会给tracker分配很小的上行带宽, 因此建议不要尝试将文件存在tracker上!!!**
 - **注意: 报文的收发必须通过已提供的 `_send_()`和`_recv_()`函数, 禁止 `import socket`**
- PClient.py (需完成下列四个函数, 四个函数均阻塞, 即未成功功能时不会返回)
 - register()
 - 功能: PClient向Tracker注册, 告知其自己拥有某个文件且愿意将其提供给其他PClient下载
 - 输入: 一个文件路径, 如当前文件夹下的alice.txt文件时即为"./alice.txt"
 - 输出: fid, 即一个文件的特定标识符, 自行设计, 不规定类型, 可以使用文件的hash值等
 - download()
 - 功能: PClient先向Tracker请求可以从哪些结点下载某个文件, 然后向这些结点请求下载

- 输入: fid, 指明一个特定文件, 类型与register()的返回值相同
 - 输出: bytes类型的数据, 其中为整个文件的内容
- cancel()
 - 功能: 取消Tracker上某个文件的注册, 此函数返回后其他PClient将无法从此处得到想要的文件
 - 输入: fid, 指明一个特定文件, 类型与register()的返回值相同
 - 输出: 根据需求自行设计, 可以无返回值
- close()
 - 功能: 取消Tracker上所有对相关文件的注册, 退出P2P网络, 不再能够收发任何报文
 - 输入: 无
 - 输出: 无
- 请在不改变, 不违背给定要求的情况下自行设计并实现上述四个PClient的功能
- 请视需要自行修改构造函数与添加其他函数
- **注意: 报文的收发必须通过已提供的_send_()和_rcv_()函数, 禁止 import socket**

给定的其他代码

- Proxy.py (请勿修改)
 - Tracker及每个PClient都会有一个唯一的proxy, 在其中会设置带宽进行传输时延的模拟
- SimpleTracker.py
 - 简单的Tracker实现的参考
- SC_Model/: 传统的Server-Client模型
- P2P_test/: P2P文件传输系统的一些测试文件

细节补充

- 在我们设定的P2P网络中, 只存在一个Tracker和若干PClient, 通过各自专属的proxy进行通信
 - **再次提醒: Tracker和传统意义上的Server不完全相同, 在Tracker中仅储存控制信息如当前网络中的每个PClient的信息(如地址)和每个共享文件的信息(如持有某个文件的PClient), 不存储实际文件。**
- Proxy
 - proxy会在一个端口上创建一个socket并通过其完成报文(若无指定端口号, 则随机绑定端口)
 - proxy中会有传输时延的模拟, 在初始化时可以传入上行和下行带宽
- PClient
 - 构造函数的参数列表

(tracker_addr, proxy=None, port=None, upload_rate=0, download_rate=0)

 - 必须指定P2P网络中的Tracker的地址
 - 如果传入了一个proxy, 该PClient将通过其进行通信
 - 如果未传入proxy
 - 传入了具体的port, 则该PClient则会尝试将proxy绑定在给定port上并通过其进行通信
 - 未指定port或指定的port已被占用, proxy将绑定在随机port上并通过其进行通信
- 基本场景描述: PClient A 加入网络并想下载文件 F
 - A初始化并被给定一个Tracker地址
 - A 向Tracker请求文件 F 的相关信息 (PClient与Tracker的交互方式自行设计实现, 不做细节要求)

- A 有文件 F 的PClient请求文件并完成下载 (PClient之间的文件分享方式自行设计实现, 不做细节要求)
- A 通过调用close()退出网络

测试场景

我们的测试将包括, 但不限于以下场景, 更多场景可见P2P_test文件夹中提供的测试代码:

- (如SimpleTest.py) Client A, B加入P2P网络, A向Tracker注册文件 $F1$, B下载 $F1$, A取消对 $F1$ 的注册, Client C 加入网络, C 下载 $F1$
 - 结果: B从A处下载文件, C从B处下载文件
- Client A, B, C, D加入P2P网络 (A有着低上行带宽, B,C,D有着高上下行带宽), A向Tracker注册文件 $F1$, B,C,D都下载 $F1$
 - 结果: 因为A的低上行带宽而B,C,D都有着高上下行带宽, B,C,D几乎在同时完成对文件的下载

评分标准

- 简单网络环境下能够正常完成运行 (30) —— 文件传输过程中不会有新节点的加入或是原有节点的退出
- 复杂网络环境下能够正常完成运行 (20) —— 文件传输过程中存在新节点的加入或是原有节点的退出
- 在不同网络环境下与传统Server-Client模型相比的效率 (20)
- 答辩(30)
 - 需要制作ppt, 必须包括以下内容, 顺序可自行调整
 - 系统设计 (Tracker, PClient、通信规则及文件传输协议)
 - 实现方法 (介绍使用的技术以及是如何使用的)
 - 项目亮点 (你最想展示的设计或实现方面的内容)
 - 测试环境描述 (建议除给定的三个测试之外, 加入自己设计的测试环境)
 - 测试结果
 - 总结 (要点回顾+未来优化方向)
- BONUS(20): tit-for-tat(见理论课课件Chapter2-3的10-11页)
 - 实现文件下载时对高上行带宽节点的优先选择(10)
 - 当一个原先互相分享的PClient的上行带宽突然下降时, 主动choke该节点并选择其他高速PClient
 - 实现对被choked的高速PClient的重新选择(10)

设计思路

- 你可以参考以下顺序来设计与实现你的项目
 - 设计Tracker上存储的具体信息(如存储整个文件/每个文件的各个文件块)
 - 设计Tracker给PClient提供的API
 - 设计Tracker与PClient之间的通信规则、报文格式
 - 设计简单环境下PClients之间的通信规则、数据报文和控制报文的格式
 - 设计策略应对复杂网络环境(如使用超时机制检测请求的结点是否已离开网络)

TIPS

- 如果不知道如何使用 `_send_()` 和 `_recv_()` 函数, 可参考udp socket中的 `sendto()` 和 `recvfrom()` 函数, 以及给定的SC模型中这两个函数的用法

- 如果使用hash函数生成fid, 建议使用hashlib, python自带的hash函数依赖于环境变量PYTHONHASHSEED, 可能导致同一内容hash出不同的结果
- **强烈建议自学并运用python的多线程知识(这将能够极大提升文件共享的速度)**
- 如果对文档或是项目有任何问题, 欢迎随时联系我们或是在下方链接中进行留言
 - 【腾讯文档】P2P文件传输系统 <https://docs.qq.com/sheet/DRVl1SUh5T1JYZ0ZN>