



Ch4. P2P原理与技术

4.1 P2P网络基本概念

4.2 混合式P2P网络（第一代）

4.3 无结构P2P网络（第二代）

4.4 结构化P2P网络（第三代）

4.1 P2P网络基本概念


What is P2P ? (Peer-to-Peer)

- 对等(网络, 计算)...; 端到端...
- 经系统间直接交换来共享计算资源和服务的应用模式
- 以非集中方式使用分布式资源来完成关键任务的一类系统和应用
 - ☞ 资源包括计算、存储、带宽、场景（计算机、人和现场）和信息等资源
 - ☞ 关键任务可能是分布式计算、数据/内容共享，通信和协同、或平台服务

◆ 典型位置：因特网边界或ad-hoc网内

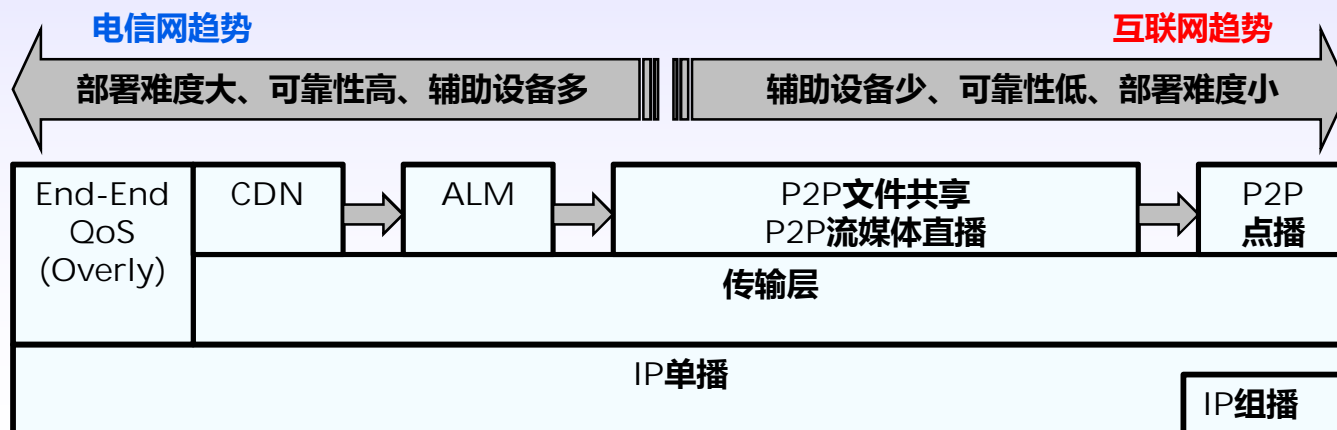
典型定义

- ◆ Intel 工作组：通过在系统之间**直接交换来共享计算机资源**和服务的一种应用模式
- ◆ A. Weytsel：在因特网周边以非客户地位使用的设备
- ◆ R. I. Granham：通过3个关键条件定义
 - 具有服务器质量的可运行计算机
 - 具有独立于DNS的**寻址**系统
 - 具有与可变连接合作的能力
- ◆ C. Shi rky：利用因特网**边界的存储/CPU/内容/现场等资源的一种应用**。访问这些非集中资源意味着运行在不稳定连接和不可预知IP地址环境下，P2P节点必须运行在DNS系统外边，对中心服务器来说具备有效的或全部的**自治**

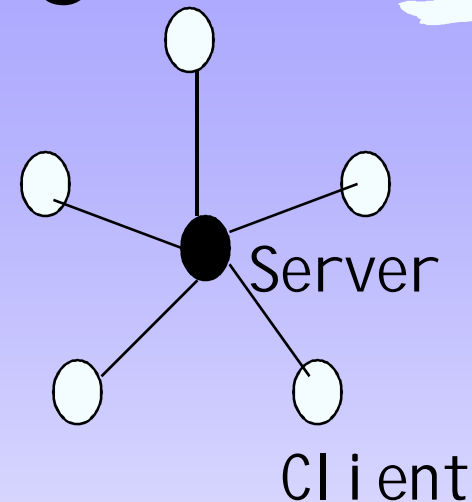
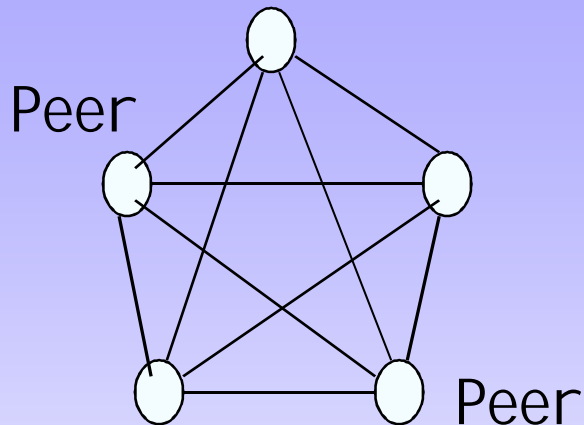
- 
- ◆ Ki ndberg: 独立生存的的系统
 - ◆ D. J. Mi l o j i c i c: 给对等组提供或从对等组获得**共享**
 - ◆ 另一种应用模式选择:
 - 相对集中式、和C/S模式
 - 纯P2P: 没有服务器的概念, 所有成员都是对等端
 - ◆ 并不是新的概念
 - 早期分布式系统: 如UUCP和交换网络
 - 电话通信
 - 计算机网络中的通信、网络游戏中的诸玩家

互联网内容分发技术演变与比较

- ◆ 单播C/S：不适合同时多用户高带宽传输；
- ◆ IP组播：需网络层设备支持，难普遍部署；
- ◆ 重叠网：提供端端质量保障，如RON/QRON，演变为改变网络层或传输层协议、多路径并加入专用设备，价值？
- ◆ ISP-CDN：成本高、部署难、大规模？
- ◆ ALM：树拓扑应用层组播，抗扰动性差、下游节点延迟大、节点间难同步，难满足用户需求？

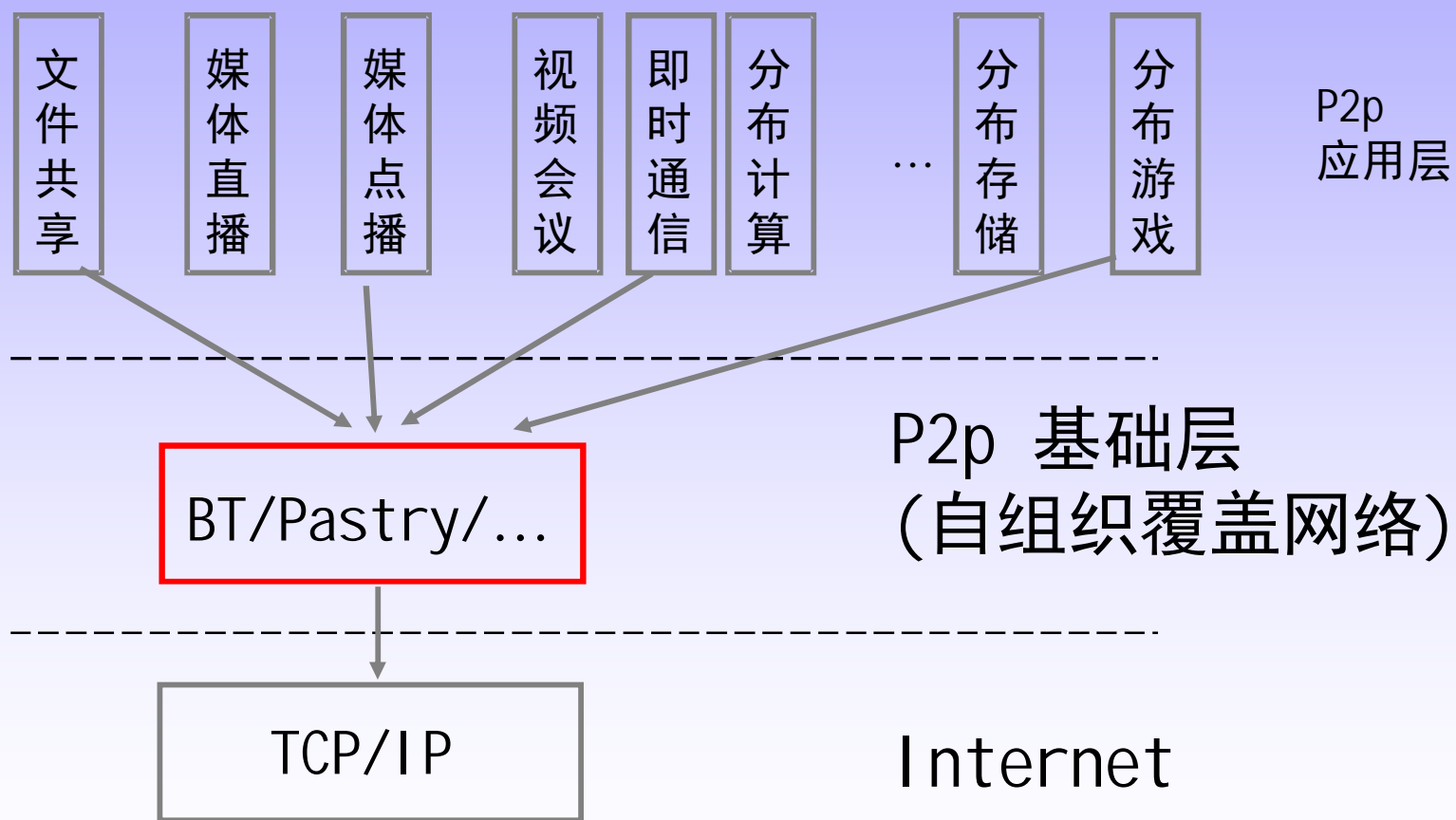


P2P 与 C/S



- ◆ 二者在结构和构成上有很大的区别
 - 管理能力、构态能力、功能（查找或发现）、组织(分层与网孔)、元素（DNS）和协议(IP)
 - C/S通常是简单的端到端通信，P2P通常要构成自己的应用层网络
- ◆ 但又无明显的边界
 - 都能运行在不同的（Internet / Intranet）平台上
 - 都能服务传统或新的应用：eBusiness eServices ...

P2P的定位



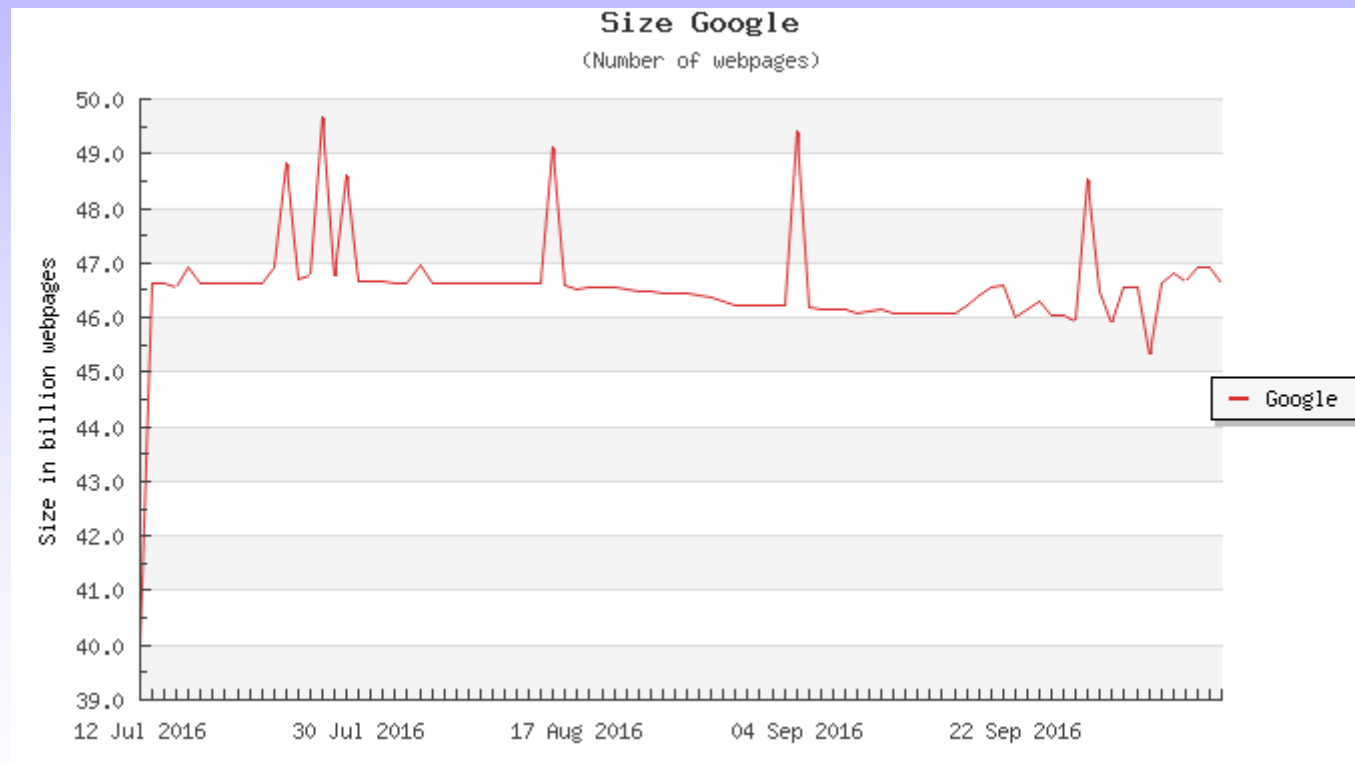
C/S 模式的挑战

单服务器或搜索引擎已不能满足或覆盖日益增长的Web内容需求

- 2×10^{18} Bytes/year Internet上增长.
- 但仅 3×10^{12} Bytes/year 可被公众利用 (0.00015%).

C/S 模式的挑战

Google 可搜索的网页





C/S

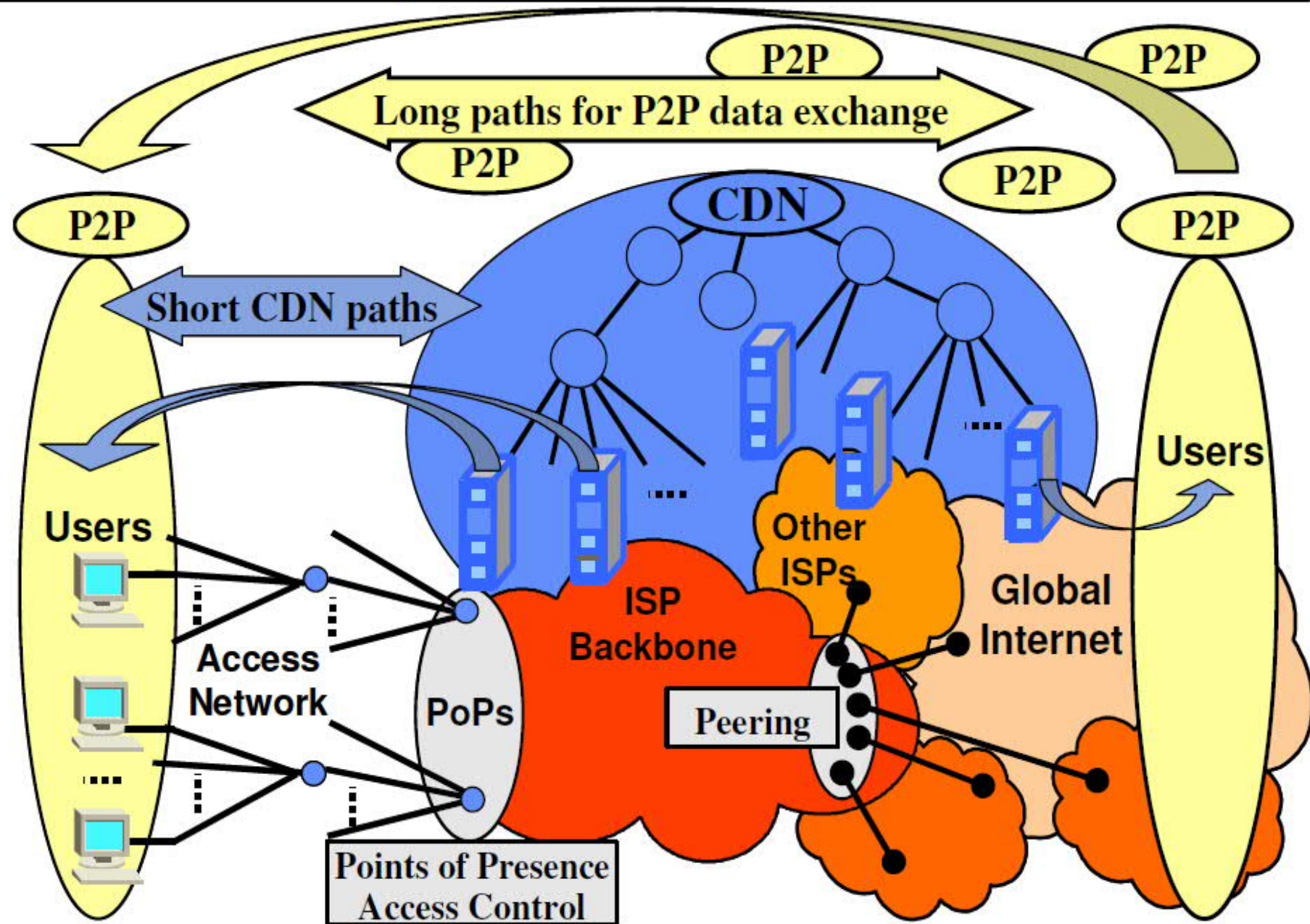
C/S 模式严重限制**可用带宽和服务**的利用

- 流行的服务器和搜索引擎已成为**流量瓶颈**
- 但许多高速网络连接的客户端却**很空闲**
- 客户端的计算能力与信息被忽视

Content Delivery Networks模式

- 服务器在因特网上分散部署（内容重复）
- 分布部署的服务器由总部中心授权控制
- Examples: Internet content distributions by Akamai, Overcast, and FFnet.
- C/S和CDN 模式都有单点失效问题

Transmission paths in CDN and P2P networks



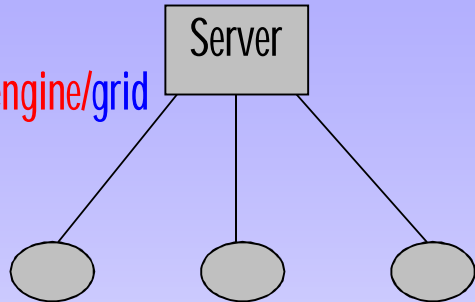
面向Peer的系统

- 既是客户端consumer也是服务器端
producer=Prosumer
- 任何时候都有加入或离开的自由
- 无限的peer diversity: 服务能力、存储空间、网络带宽和服务需求
- 挑战与机遇: 开放的广域无中心分布系统



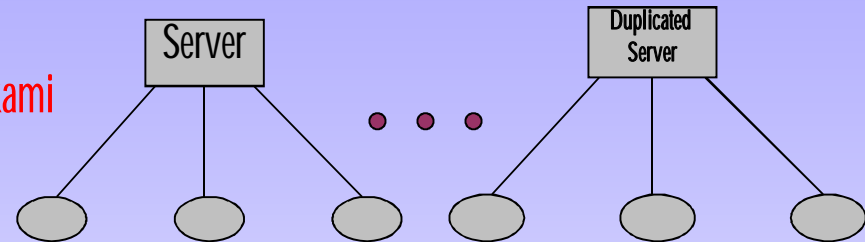
C/S模式

a search engine/grid



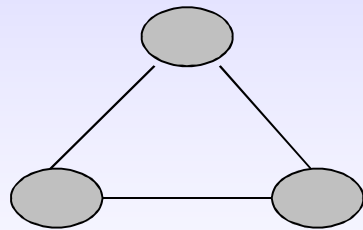
Content Delivery Networks

e.g. Akami



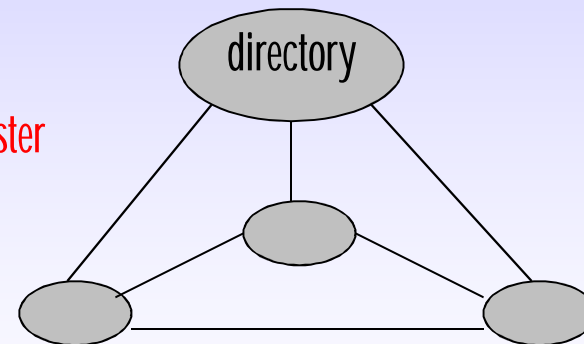
Pure P2P

e.g. Freenet & Gnutella



Hybrid P2P

e.g. Napster



P2P 的目标与优势

- ◆ 只要不存在网络的物理断开，目标文件总是可以找到！
- ◆ 信息可扩展：往P2P系统加入更多内容将不影响其性能！
- ◆ 系统可扩展：加入或离开，将不影响P2P 系统的性能！

P2P 的目标与吸引力

- ◆ P2P是一类发挥互联网边缘资源（存储、处理能力、内容、带宽、用户现场）可用性的应用
- ◆ 每个参与者（进程）既是客户端也是服务器：
 - 你的PDA可以存放部分音乐目录
 - 你的PC可以存放部分音乐仓库
- ◆ 简化地依赖个人设备和子网（去中心服务器）
- ◆ 非脆弱的健壮性（无单点故障）
- ◆ 柔软性/快速恢复（内建冗余）
- ◆ 抗DOS攻击（无中心服务器）
- ◆ 更高的可扩展性
- ◆ 改进的高峰请求服务（提出需求的设备越多，意味着服务器资源也越多）

P2P的效果

◆ 巨大的扩展力

- 通过低成本交互来聚合资源，导致整体大于部分之和

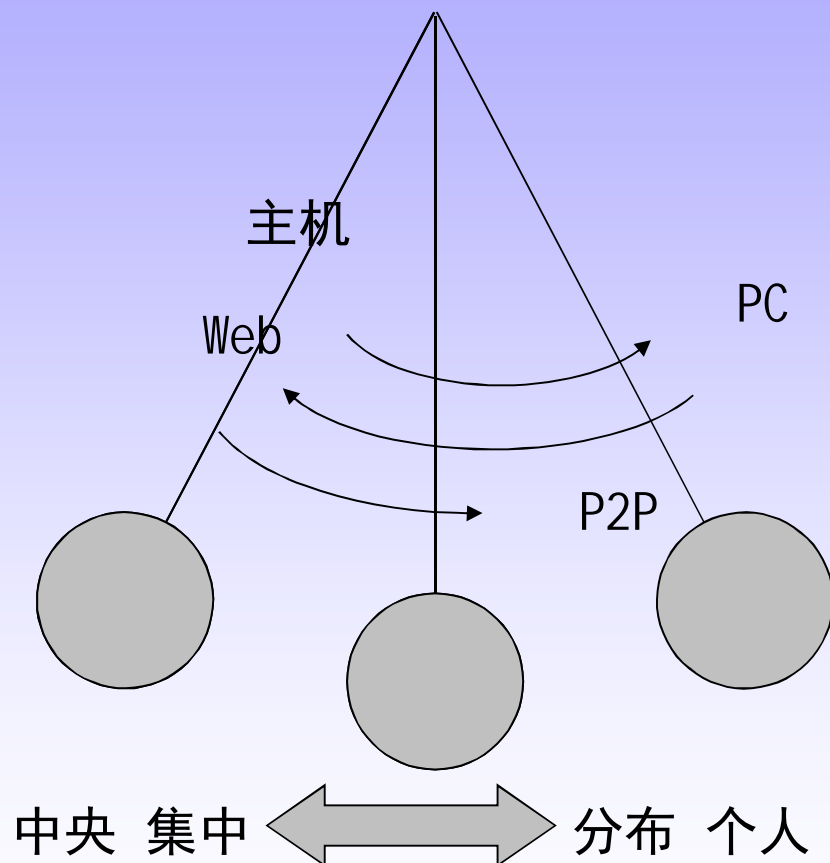
◆ 低成本的所有权和共享

- 通过使用现存的基础设施、削减和分布成本达到

◆ 匿名和隐私：

- 通过允许对等端在其数据和资源上很大的自治控制达到

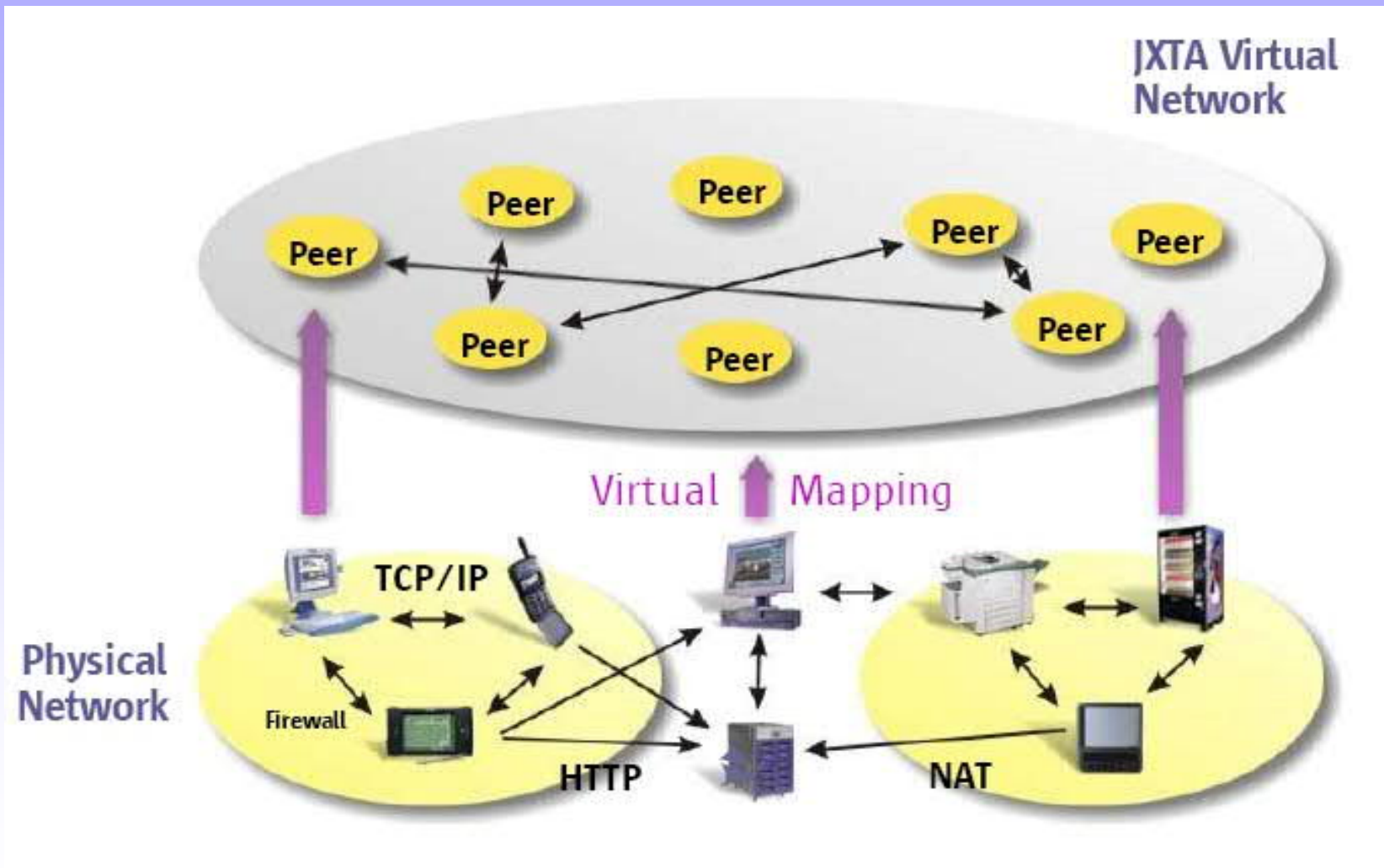
P2P-从集中向分布的演化

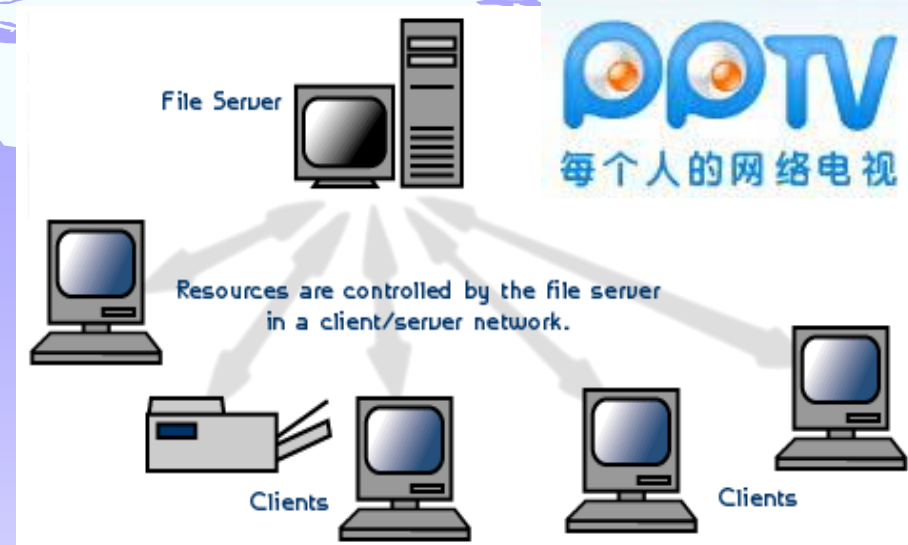
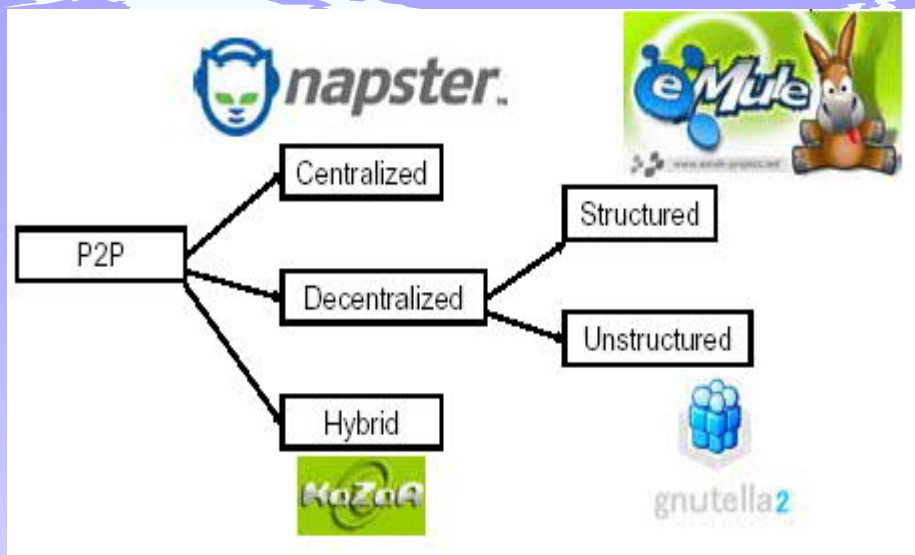


- ◆ 将每一个 PC 变成服务器
- ◆ 适合自组织 ad-hoc组工作
- ◆ 推动采用 IPv6, 用户直接连接网络
- ◆ IPv6提供无服务器DNS
- ◆ 开发者的平台

充分发挥互联网无所不在的优势

应用层重叠网络





即时通讯软件



下载软件



P2P游戏等

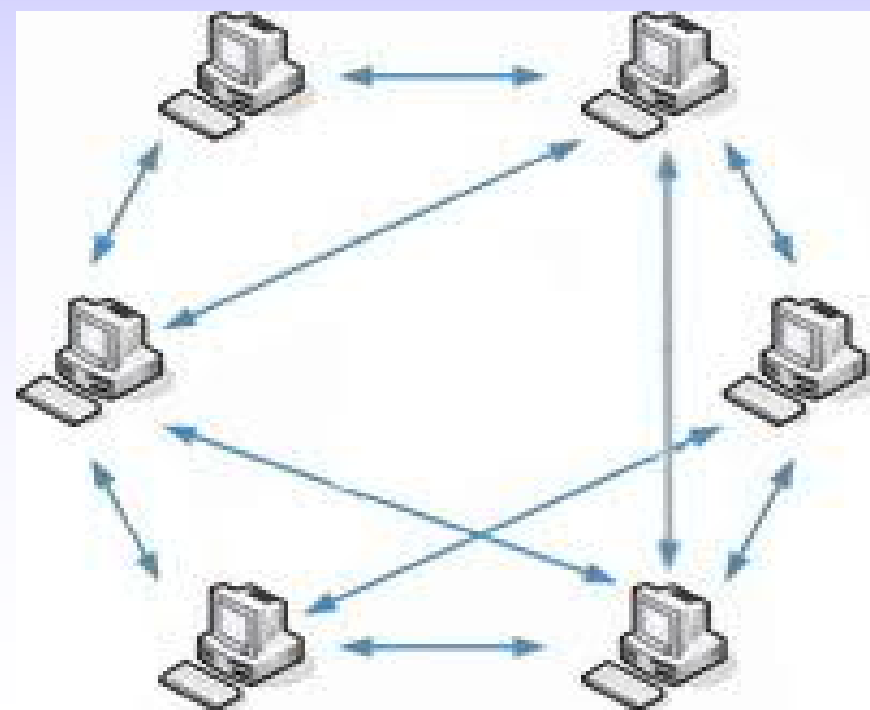
流媒体



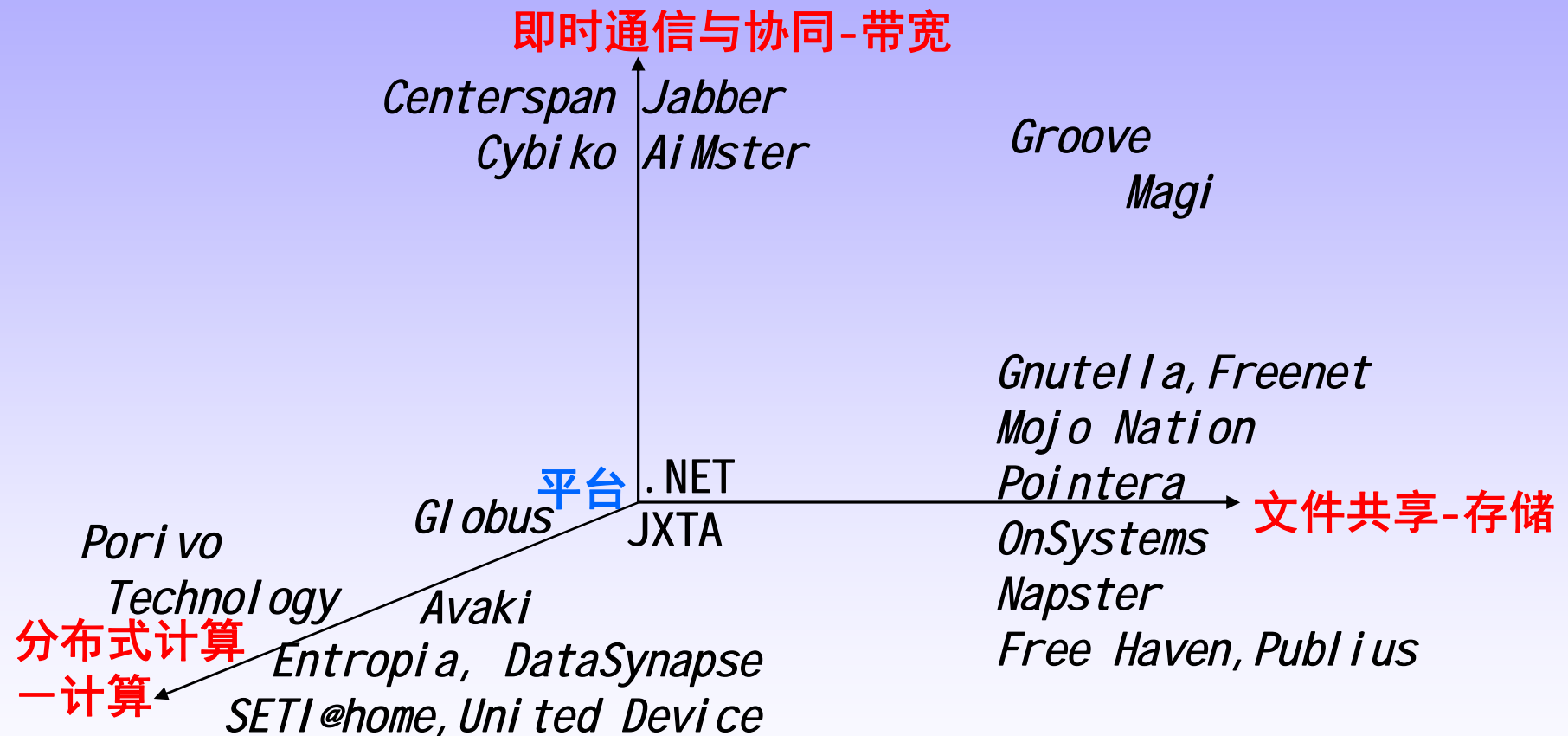
匿名访问



科学研究



P2P应用的多维视图



4.2 混合式P2P网络（第一代）

4.2.1 集中目录模式Napster

➤ C/S集中目录查询，P2P下载

- ☞ Peers连接到能提供共享内容的中心目录上，匹配请求与索引
- ☞ 文件直接在两个Peers间交换

➤ 需要一些可管理的设施

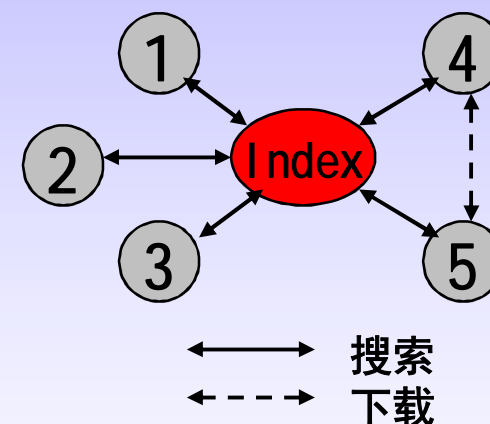
- ☞ 目录服务器：记载群组所有参加者的信息

➤ 限制了规模的扩大：

- ☞ 大量用户增加→大量请求→大服务器→存储器

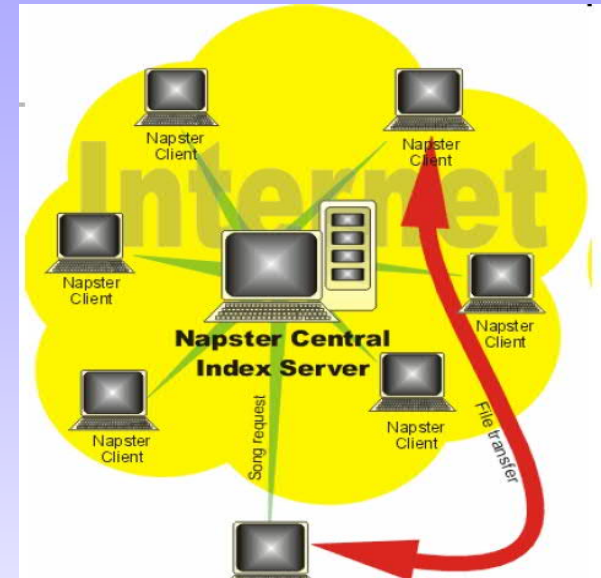
➤ 然Napster经验表明：

- ☞ 除开法律问题外，该模式还很有效和强大
- ☞ 1999.12被多家唱片公司起诉、并败诉

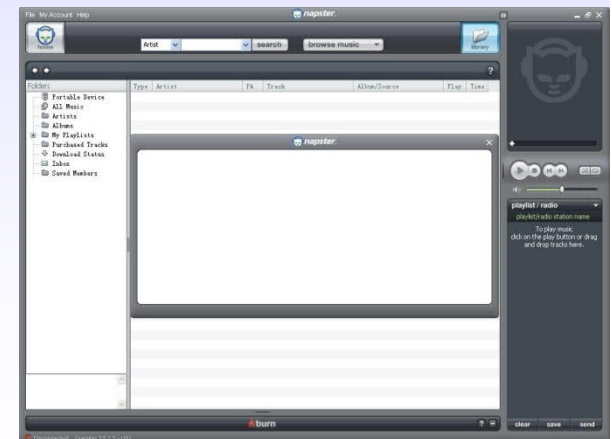


P2P网络先驱 Napster

- ◆ 1999, 18岁Shawn Fanning开发
 - 让音乐迷交流MP3文件
 - 服务器只提供索引, 无任何歌曲, MP3文件在Peers自己的硬盘上
- ◆ 第一个在互联网上不经过服务器直接交换文件的应用体系
 - 发布半年后, 吸引到5000--6100万注册用户
 - 然其初衷只是想能在Boston的东北大学校园与Virginia的朋友共享MP3歌曲

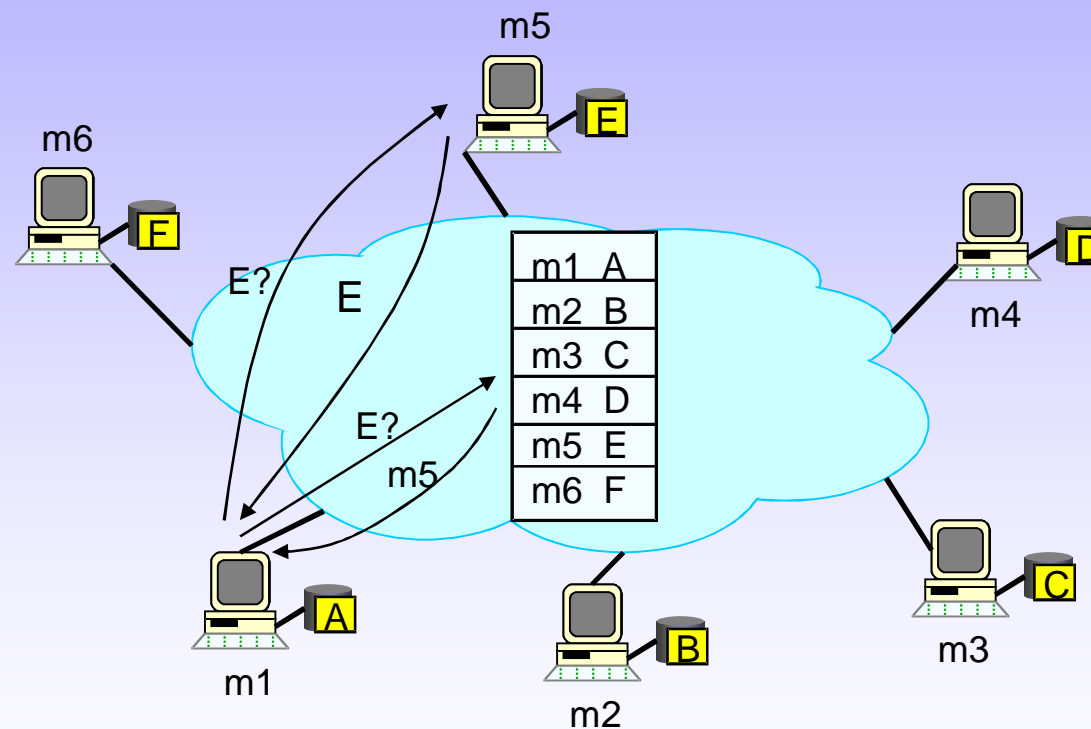


Napster结构



Napster界面

Napster: Example



Napster原理



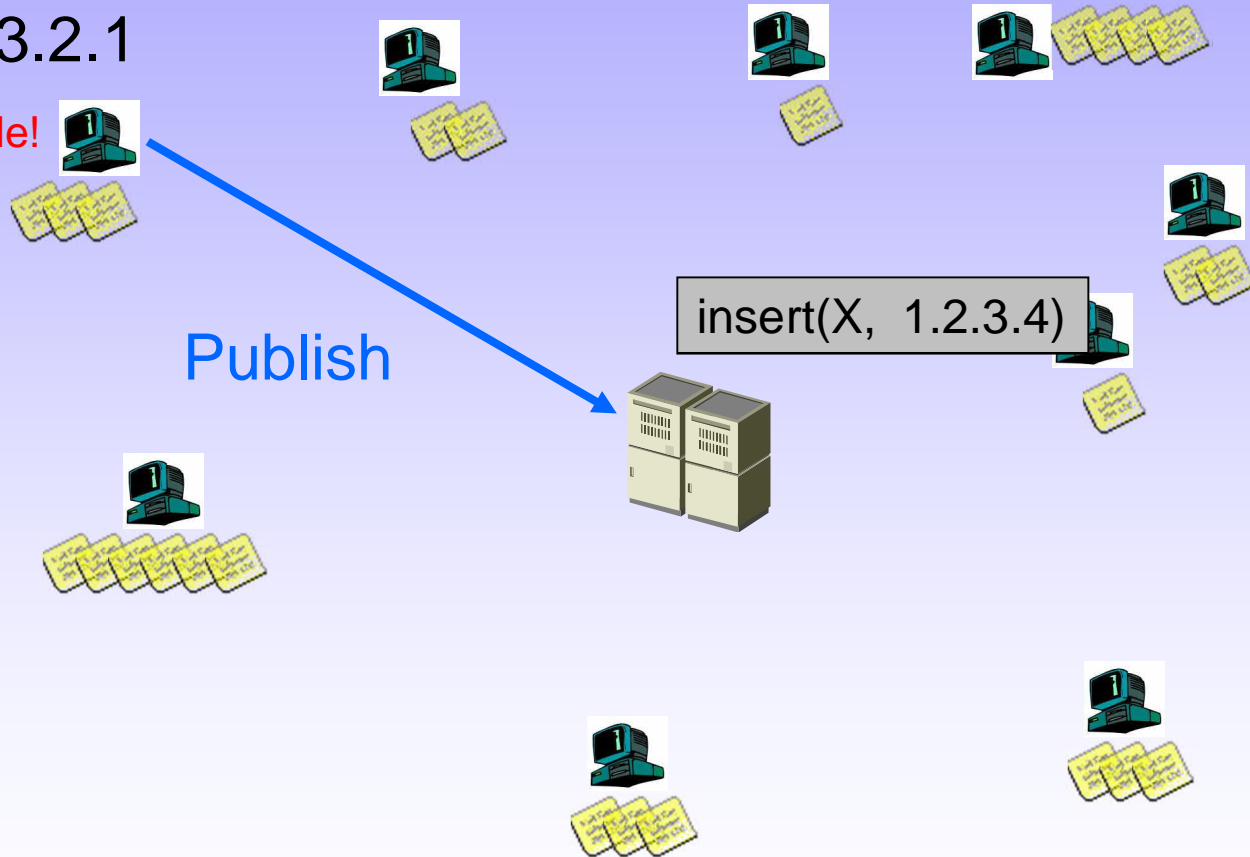
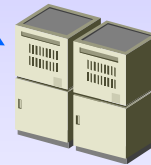
4.3.2.1

I have 天堂.mp3 file!

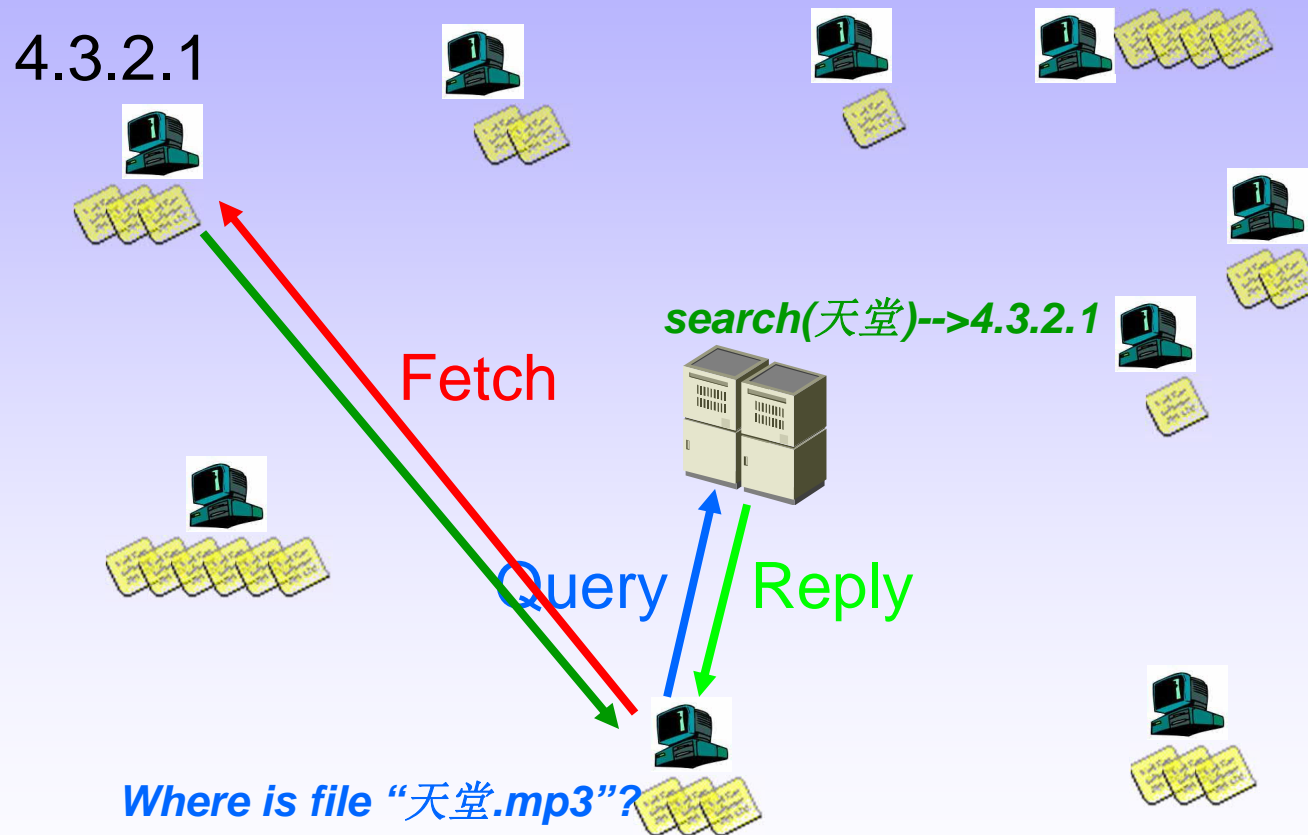


Publish

insert(X, 1.2.3.4)



Napster原理



Napster官司及缺点

◆ 官司

- 1999. 12. 7, 美国唱片协会代表7大唱片公司指控Napster侵犯音乐版权, 要求法院关闭该公司并赔偿1亿美元;
- 2000. 2月, 旧金山第九巡回上诉法院的3名法官裁决, 认为Napster一直知道并纵容用户侵权, 但没有应要求立即关闭Napster, 而是把初判送到低一级地方法院
-
- 2001. 7. 12, 法官Patel 命令Napster继续关闭直到它可以证明能够有效阻止对版权文件的使用。

◆ “魔鬼”钻出了“魔瓶”：

- Napster背后的技术和思想给互联网带来的极大影响, 而魔瓶已经打破, 唱片经销方式被彻底改变, 并带来互联网新的long Tail 现象

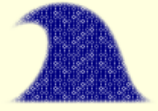
◆ Napster的缺点：

- C/S的单点故障、系统瓶颈、可扩展性低
- 未考虑不同用户的能力差异、无鼓励机制
- 版权问题

4.2.2 分片优化的Bi tTorrent

◆ BT故事

- 2002. 10, 穷困潦倒的Bram Cohen发明BT, 免费软件企业家John Gi lmore帮助他解决了部分生活费
- 2003初, 在Internet2(连接200多所美国大学的Abi l ene主干)用BT发行一个新版的Li nux和日本卡通, 速度比ADSL快3500多倍, 10月流量超过Internet2流量的10%
 - ☞ 但2003. 9, Bram Cohen还在用一张信用卡的免费透支来偿还另一张的账单
- Valve软件公司的董事Gabe Newell正在游戏分发网络, 在Seattle给它一个职位
- BT既指一个混合式P2P网络, 也指其对应的协议及支持该协议的应用软件--Bi tTorrent/Bi tComet/Bi tSpi ri t/FI ashBT
- 中国BT网站和搜索引擎: BT@Chi na联盟/冰鱼BT站/影视帝国/教育网总站5Q/好123网址之家



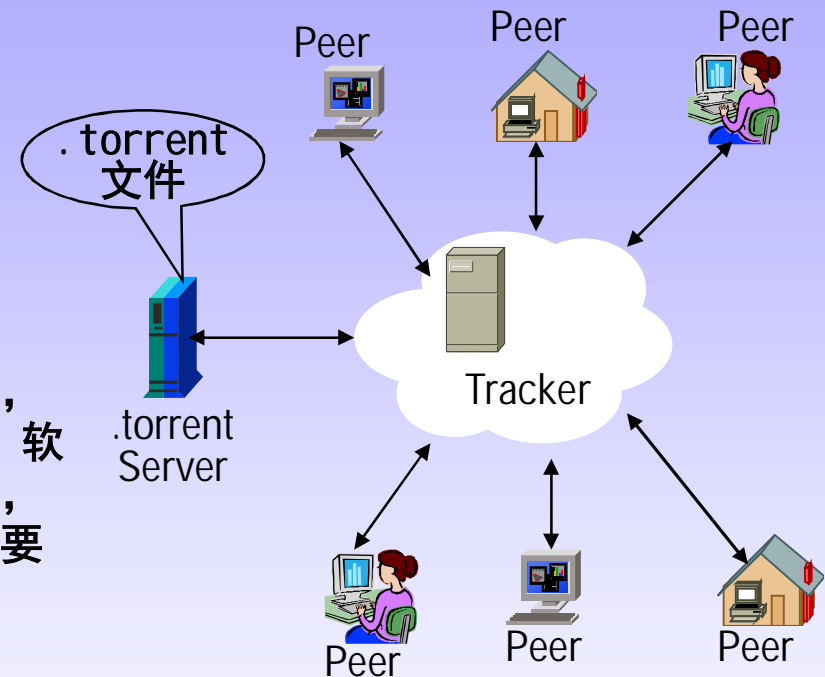
BitTorrent原理

◆ 构成（4大部分）

- BT网站+.torrent文件服务器+Tracker+BT用户

◆ 各部功能

- Seeds：拥有整个文件的用户
- BT网站提供BT种子文件.torrent
 - ☞ 种子文件的一个子集，用户在其返回.torrent中再选择种子文件
 - ☞ 内含.torrent上传日期，文件名，大小，种子数，在tracker注册的下载用户数，软件运行的OS，上传.torrent文件的用户，相关网页的连接 + Tracker的位置，所要下载文件的Hash值
- Tracker：用户信息维护者
 - ☞ 跟踪所有下载同一文件的用户，构成1独立子网，实时分发所有用户信息给每个Peer
 - ☞ 和用户之间用HTTP协议交互。
 - ☞ 用户告诉要下载的文件、自己的端口号等
 - ☞ Tracker告诉下载同样文件的其它用户信息
 - ☞ 收集和统计上传和下载的相关信息



BitTorrent 结构

我为人人，人人为我

◆ 上载

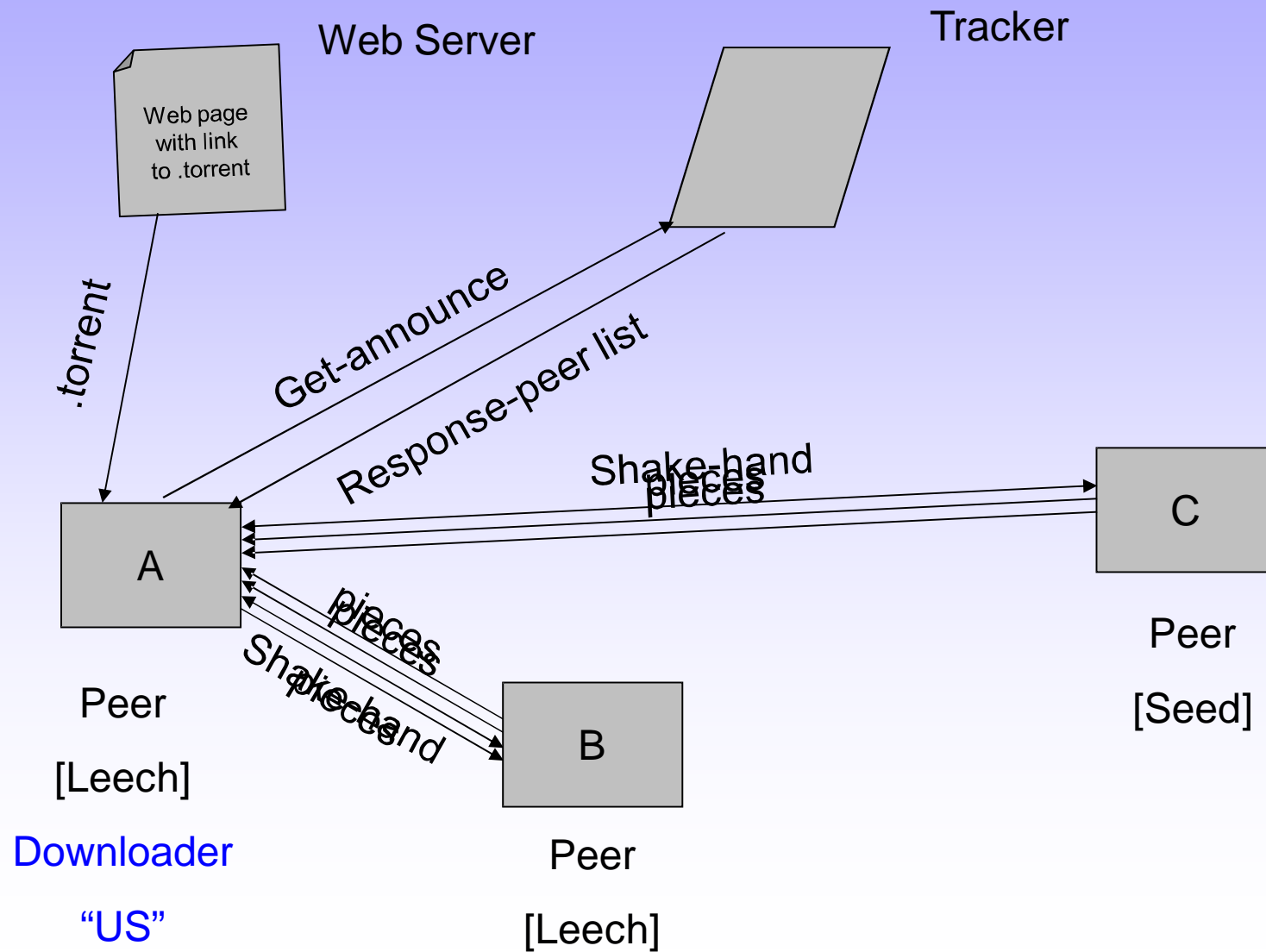
- 某Peer想要共享文件或目录，则为该文件或目录生成一“种子”文件（或元信息：含该文件或目录名+用户的URL信息）
- 然后把该“种子”上传到BT服务器上（或Tracker）

◆ 下载

- 需下载的Peer到Tracker上找到所需种子
- 根据种子信息进行下载

1. 用户通过BT网站搜索文件，得到.torrent文件列表
2. 用户选择列表中的一项或多项
 - 每个被选.torrent文件会启动一项下载任务，
 - 通常.torrent会指向该文件对应的Tracker，
 - 而Tracker会把一部分用户信息给请求者
3. 用户根据Tracker用户信息
 - 与其它用户建立连接
 - 从它们下载文件分片，也提供分片
4. 高速高效
 - 并不总是和最初邻居交换分片，而是每隔一段时间从Tracker获得新的邻居
 - 采用“阻塞算法”主动停止那些对自己无贡献的邻居，寻求更好的邻居

BT 的结构





BitTorrent的分片与分发

◆ 分割分片

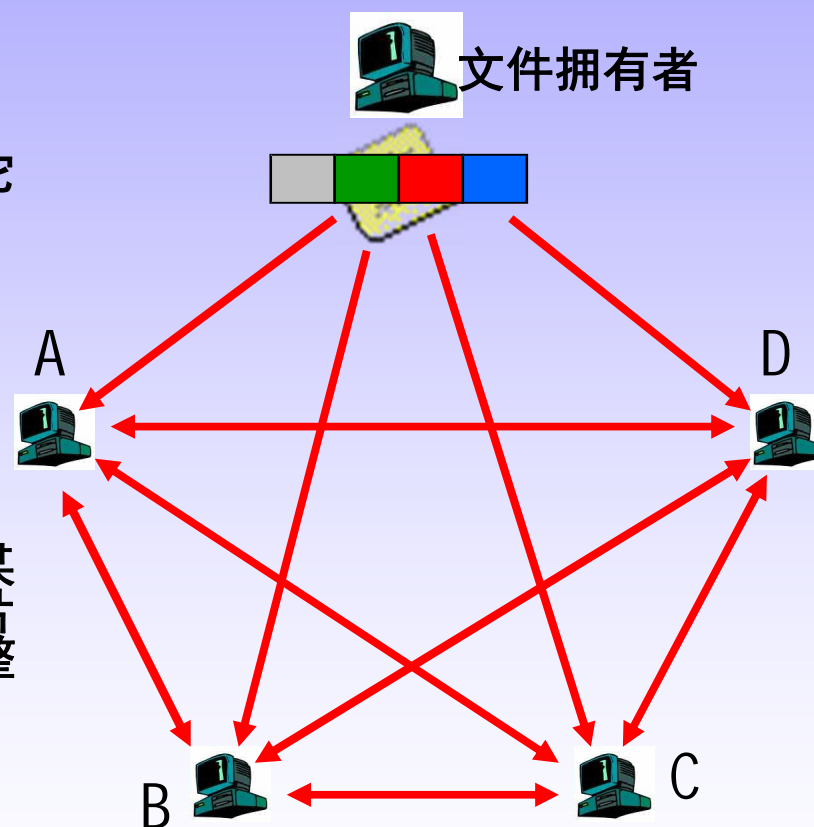
- BT把文件分割成相同大小的块，典型256KB，为了并发下载再分为16KB的子分片
- 用SHA-1对每个分片或子分片生成**校验值或文件块ID**；保存在.torrent中，
- 只有在验证其唯一性完整性后才通知其它peer自己拥有该片

◆ 流水分发

- 在TCP之上，可流水地同时发送多个请求，通常5个，以避免两个分片间的延迟

◆ 分片选择的阶段规则

- **最初：随机选择**一个分片
- **分片下载中：整分片优先**：一旦请求了某个分片的子分片，则该分片剩下的子分片优于其他分片被请求，以尽快获得一个整分片
- **文件下载中间阶段/平稳期：最少者优先**：选择Peers拥有最少的分片，最新的分片（非常关键）
- **最后：尽快取消**。为防止最后阶段的潜在延迟，多发请求；一旦得到最后子分片，就向其它用户发出Cancel消息。





Bi tTorrent原理

◆ 协议

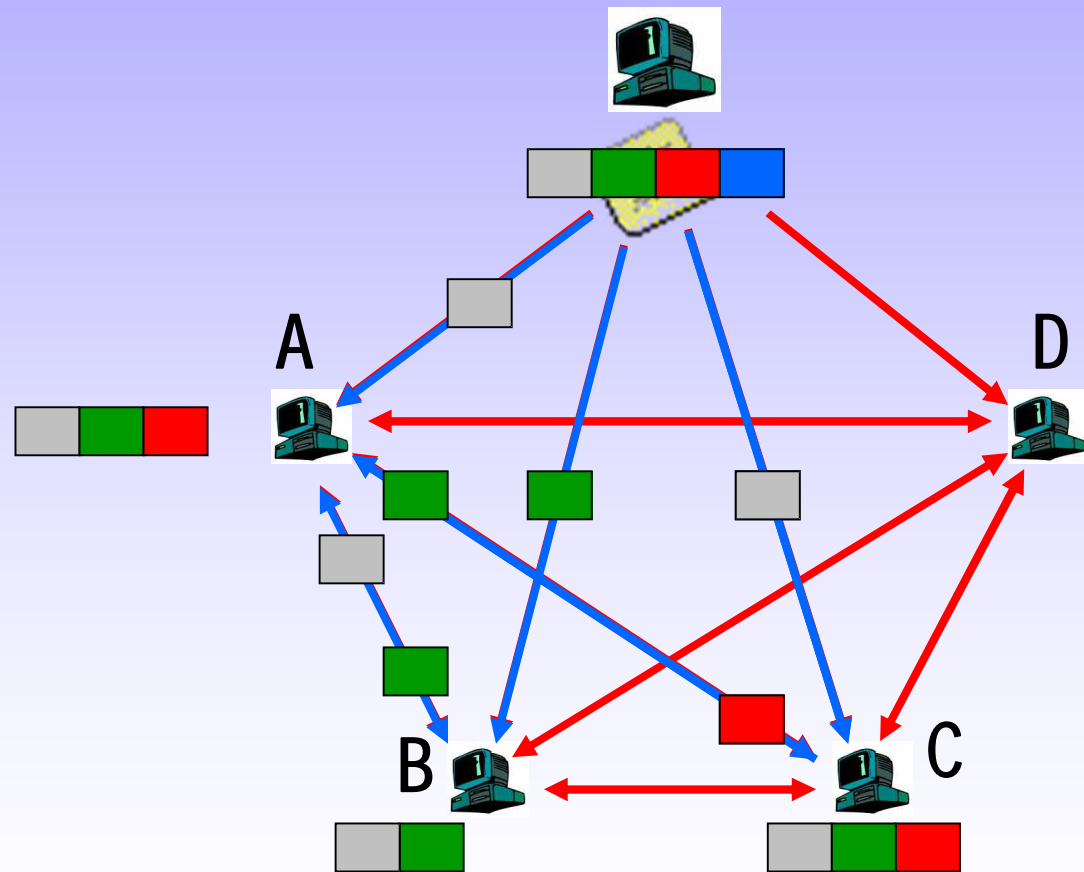
- 种子文件**上传****下载**、Peers和Tracker间通信都是使用**HTTP**协议
- 各Peers间通信使用**Bi tTorrent Peer协议** (TCP)

◆ 问题

- Tracker的**单点失效**
- 未对Peers**身份认证**

◆ 开发

- Bi tTorrent协议公开，任何人可开发服务器端或客户端
- 国内流行Bi tComet，用**Python**语言开发



BitTorrent的阻塞算法

- ◆ Tracker并不集中分配资源，而由用户控制
 - Peer尽可能提高自己的下载效率
 - Peer根据下载方决定对其上传回报（tit-for-tat）针锋相对
 - ☞ 对合作者，提供上传服务
 - ☞ 对投机者，阻塞对方，暂时拒绝上传服务
- ◆ 阻塞算法（Choking algorithm）
 - 经济学背景-Pareto efficient: 当系统中资源配置已达到这样一种境地：任何重新改变资源配置的方式，都不可能使一部分人在没有其它人受损的情况下受益。
 - 每个用户一直保持4个邻居的疏通。每个20秒（10+10）轮询，每隔10s决定阻塞谁，并保持该状态10s，10s足够TCP调整传输率到最大。（一个新的用户，没有下载任何分片，怎么办？）
 - 最优疏通（Optimistic unchoking）：不管其下载速率如何，每隔30秒重新计算哪个连接应该是最优疏通，这个最优unchoking保留，其他更换以便得到更好的下载速率。30s足使上传最大

优缺点总结

- ◆ 拓扑结构：服务器仍然是网络的核心
- ◆ 底层协议：全部使用TCP，限制了链接的Peer数量
- ◆ 查询与路由简单高效：
 - Napster和BT的用户访问服务器；服务器返回文件索引或种子文件；用户再直接同另一Peer连接
 - 故路由跳数为 $O(1)$ ，即常数跳
- ◆ 容错、自适应和匿名性
 - 服务器单点失效率高
 - 自组织和自适应主要依靠服务器
 - 服务器的存在使匿名性实际不可能
- ◆ 用户接入无安全认证机制