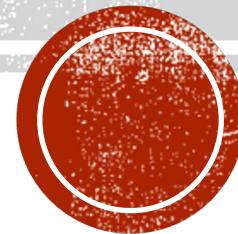


# 互联网XX 内容传输

张喆

[zhezhang@njupt.edu.cn](mailto:zhezhang@njupt.edu.cn)

通信与信息工程学院



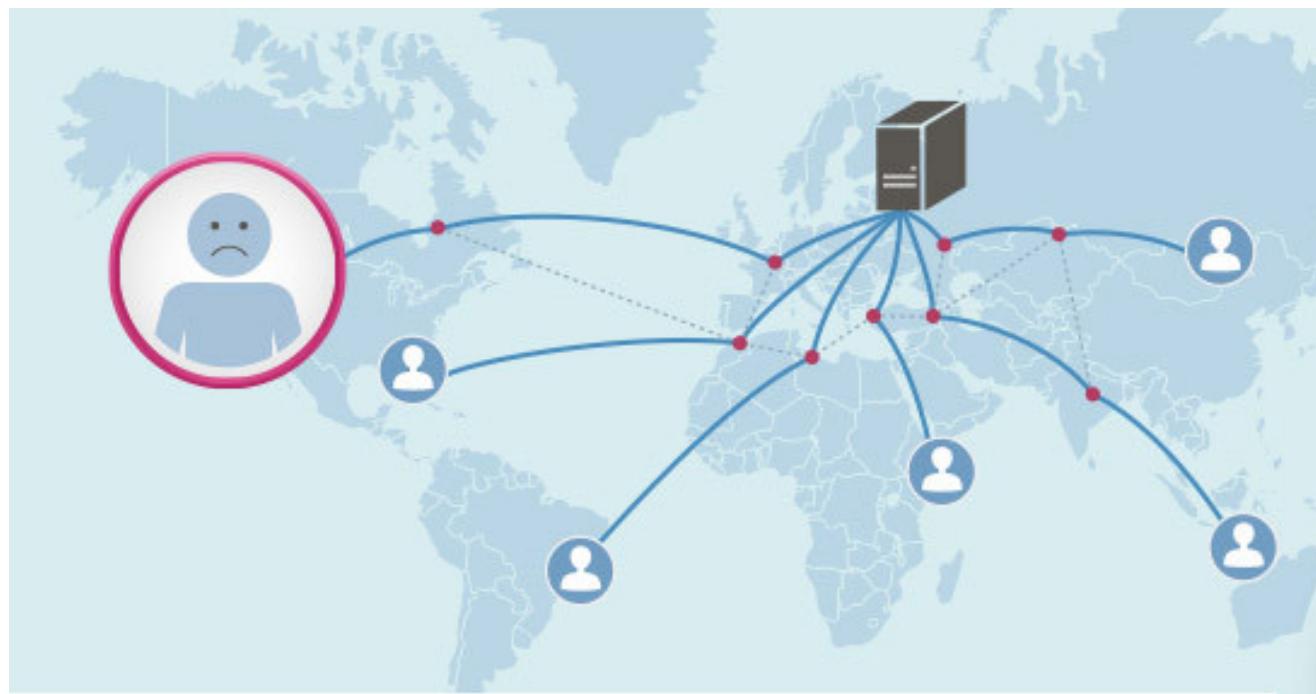
# 目 录

- 1、互联网内容与流量
- 2、多媒体内容
- 3、Server Farms and Web Proxies
- 4、内容分发网络
- 5、对等网络P2P



# 内容分发网络

- 传统C/S架构的局限：
  - 无法满足内容在全球范围内的高效传输



# 内容分发网络

- CDN (content delivery network) : 一种分布式网络架构，通过在全球范内部署分布式服务器节点，将内容缓存在离用户最近的节点上，来提供高效的内容传输和交付服务。



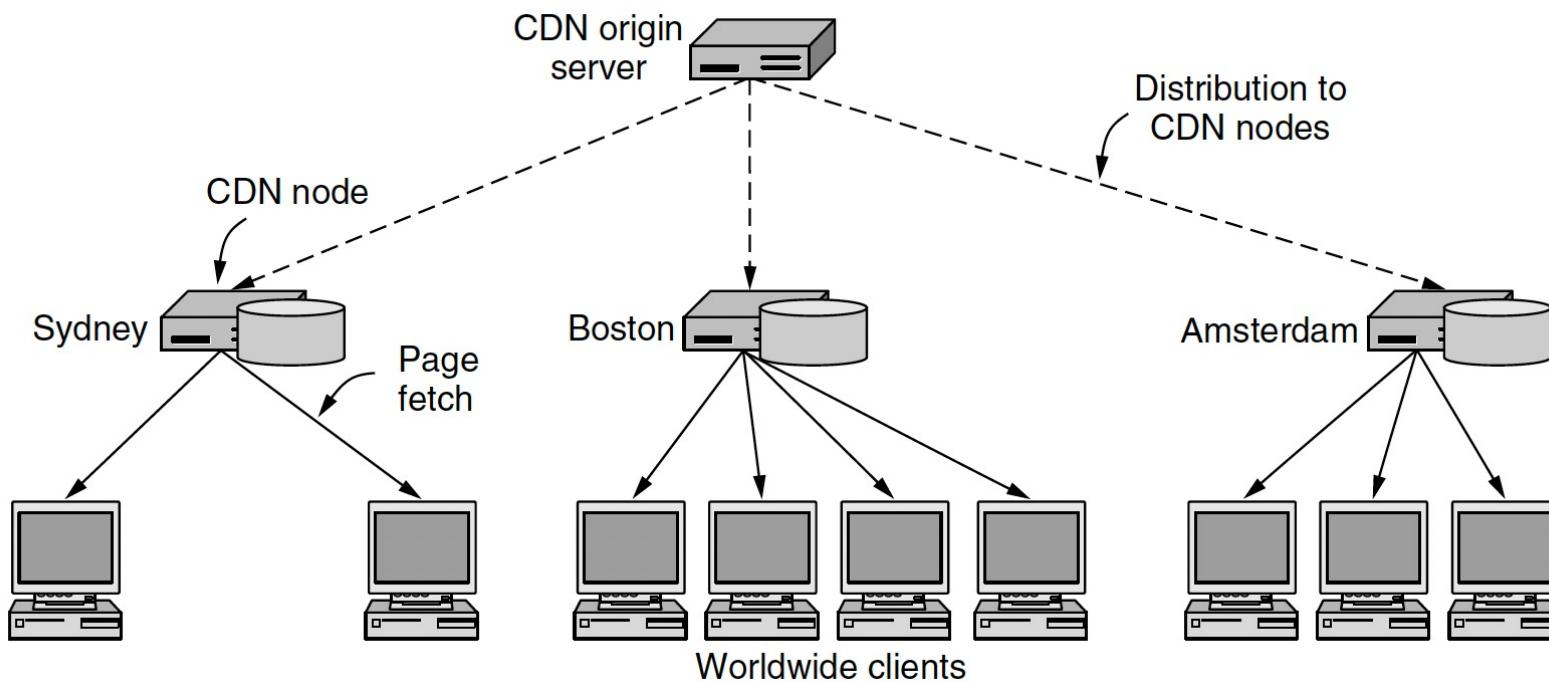
# 内容分发网络

- 如何设计CDN架构呢?
  - Straightforward way: Distribution tree
    - 简单、直接



# 内容分发网络

- CDN distribution tree



# 内容分发网络

- CDN distribution tree的好处：
  - 可扩展性：只需要增加子节点的数量、层数即可实现扩展
  - 提升内容传输效率：通过从最近的CDN服务器获取内容来降低传输时延。
  - 将低网络总负载：通过合理部放CDN服务器可将大量流量限制在网络边缘，进而降低骨干网负载。



# 内容分发网络

- 如何让用户能够使用CDN的分布式节点呢?
  - Idea is cheap, show me the code.



# 内容分发网络

- 如何让用户能够使用CDN的分布式节点呢?
  - 方案一：采用proxy server的形式
  - 方案二：采用镜像mirroring的形式
  - 方案三：采用DNS重定向的形式



# 内容分发网络

- Web proxy形式：通过配置，使用户将CDN节点当作web proxy来使用。
  - 存在以下三个问题：
    - 用户在同一网络中可能归属于不同机构
    - 存在多个CDN节点，但用户只能使用一个作为自己的web proxy
    - Web proxy由用户配置，无法确保内容分发的最优性，缺少网络优化。



# 内容分发网络

- 镜像形式：将原服务器上的内容复制到CDN节点，并通过原服务器网页中的链接指向不同的镜像站点。
  - 分布在不同网络中CDN节点称为镜像站点（mirrors）
- 存在的问题：需要依靠用户自己来选择节点进行分发

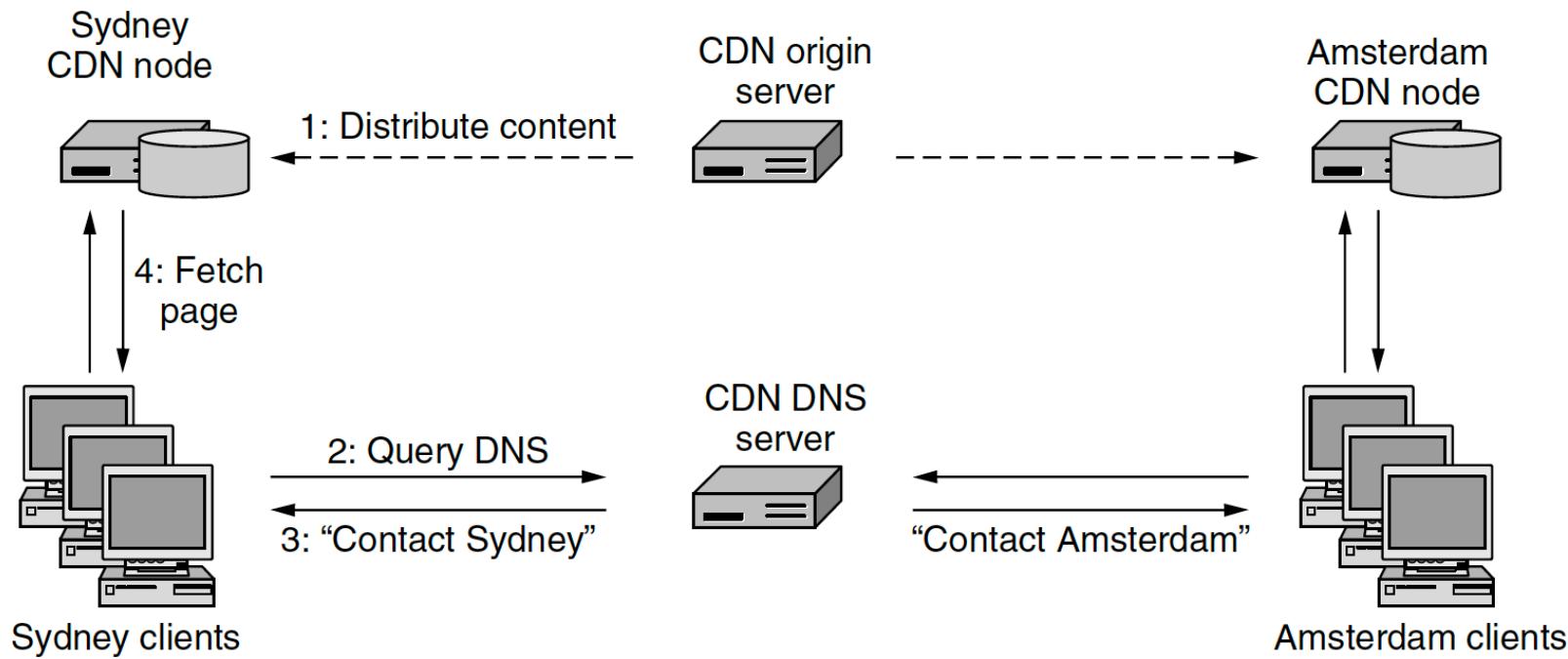


# 内容分发网络

- 为了克服前两个方案的缺点：
  - DNS重定向形式：通过使用DNS来将用户请求重定向到最佳的CDN节点。
    - 根据用户IP来返回不同的CDN节点IP。



# 内容分发网络



# 内容分发网络

- CDN的核心问题：如何将用户与最佳节点匹配？
  - 需要考虑以下两个因素：
    - 网络距离：short and high-capacity network path
    - 节点负载：考虑当前节点的负载
      - E.g., 满负载节点响应速度会下降，此刻不如选择距离稍远但负载较轻的CDN节点。



# 内容分发网络

- 商业实例：
  - Akamai：第一家大型CDN运营商



# 内容分发网络

- CDN带来的好处：
  - 对ISP：将CDN部署在ISP网络内部可以有效减少upstream网络带宽开销（ISP需要付费）。
  - 对内容提供商（content provider）：减轻服务器负载，提升内容分发效率。
  - 对用户：降低内容传输时延。
  - 对CDN运营商：make money



```
<html>
<head> <title> Fluffy Video </title> </head>
<body>
<h1> Fluffy Video's Product List </h1>
<p> Click below for free samples. </p>

<a href="koalas.mpg"> Koalas Today </a> <br>
<a href="kangaroos.mpg"> Funny Kangaroos </a> <br>
<a href="wombats.mpg"> Nice Wombats </a> <br>
</body>
</html>
```

(a)

```
<html>
<head> <title> Fluffy Video </title> </head>
<body>
<h1> Fluffy Video's Product List </h1>
<p> Click below for free samples. </p>

<a href="http://www.cdn.com/fluffyvideo/koalas.mpg"> Koalas Today </a> <br>
<a href="http://www.cdn.com/fluffyvideo/kangaroos.mpg"> Funny Kangaroos </a> <br>
<a href="http://www.cdn.com/fluffyvideo/wombats.mpg"> Nice Wombats </a> <br>
</body>
</html>
```

(b)

**Figure 7-69.** (a) Original Web page. (b) Same page after linking to the CDN.

# 内容分发网络

- Flash crowds: 是指突发性的大量用户同时访问一个网络资源或服务的现象。这种情况通常发生在某个特定事件、新闻、热门活动或突发事件引起广泛关注的时候。
- 特点：
  - 突发性
  - 高峰流量
  - 临时性



# 内容分发网络

- 应对策略：
  - 使用CDN
    - CDN可以在全球范围内提供内容分发
    - 小站点可以在flash crowds时使用CDN来度过高峰期



# 目 录

- 1、互联网内容与流量
- 2、多媒体内容
- 3、Server Farms and Web Proxies
- 4、内容分发网络
- 5、对等网络P2P



# 对等网络 (P2P, PEER-TO-PEER NETWORKS)

- CDN虽好，但不是任何人都可以部署的起CDN来分发内容。
- 如何应对？
- Solution: Peer-to-Peer networks (P2P)



# 对等网络 (P2P, PEER-TO-PEER NETWORKS)

- P2P: 是一种计算机网络通信模型, 其中计算机之间平等地共享资源和服务, 而不依赖于集中式服务器。在P2P网络中, 每个计算机都可以同时充当客户端和服务器的角色, 可以共享自己的资源 (如文件、带宽、计算能力等) 并从其他节点获取资源。



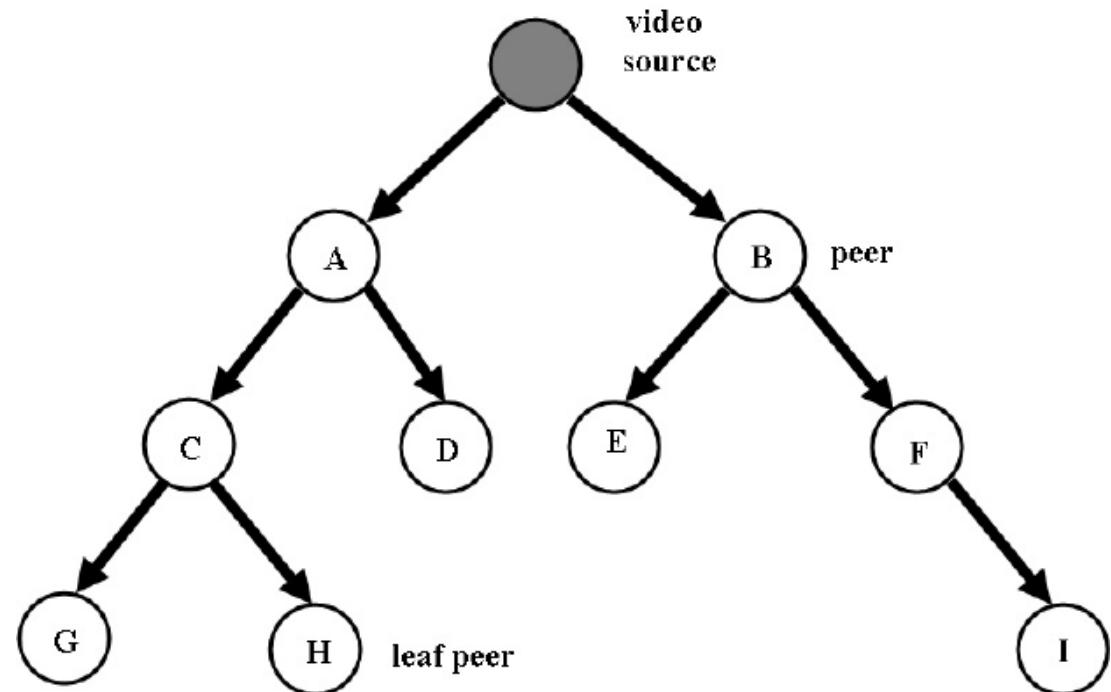
# 对等网络 (P2P, PEER-TO-PEER NETWORKS)

- 假设在一个P2P网络中有 $N$ 个用户，每个用户带宽为1 Mbps，则该P2P网络的总上传容量为 $N$  Mbps，总下载速率也为 $N$  Mbps。每个用户都可以同时上传和下载。



# 对等网络 (P2P, PEER-TO-PEER NETWORKS)

- 假设一个P2P网络按照二叉树拓扑构造
  - 任一非叶子节点向其余两个节点发送数据，文件被切为1000份。
  - 大致有 $N/2$ 非叶子节点，则该二叉树上传带宽大致为 $N/2$  Mbps，下载带宽大致也为 $N/2$  Mbps (转换叶子节点与非叶子节点关系)。



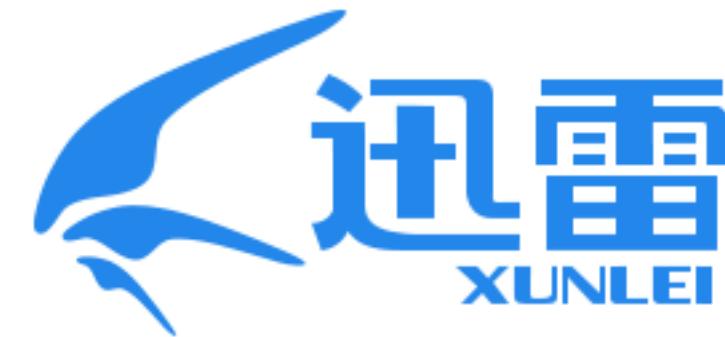
# 对等网络 (P2P, PEER-TO-PEER NETWORKS)

- P2P具有自缩放特点：网络容量随着用户数量的增加而增加。
- E.g., C/S架构下，站点有100台服务器，每台服务器10 Gbps带宽：
  - 如果用户带宽为1 Mbps，那么服务器的高带宽并不能提升用户体验；
  - 如果用超过100万个用户，那么服务器将无法为每个用户都提供高速访问。



# 对等网络 (P2P, PEER-TO-PEER NETWORKS)

- P2P协议: bitTorrent, DHT (distributed hash table)



# 对等网络 (P2P, PEER-TO-PEER NETWORKS)

- BitTorrent:
  - 需要解决三个问题：
    - 一个peer节点如何找到有它想要的内容的peer节点
    - 内容如何被peer节点复制来为所有人提供高速下载
    - Peer节点如何互相鼓励来将内容上传、下载给其他人呢



# 对等网络 (P2P, PEER-TO-PEER NETWORKS)

- 针对第一个问题 (本质是因为并非所有节点都有全部内容) :
  - BitTorrent为每一份内容提供一个说明, 称为torrent (远比内容小)
  - Torrent被peer节点用来验证所下载内容的完整性
  - 其他用户想要下载该内容必须首先获取torrent



# 对等网络 (P2P, PEER-TO-PEER NETWORKS)

- Torrent (种子文件) : 本质是一个特殊格式的文件, 包含两种关键信息
  - Name of a tracker
  - A list of chunks
    - 典型的chunk size为64KB到512KB
    - 每一个chunk赋予一个hash (160 bit SHA-1)

种子文件就是被下载文件的“索引”



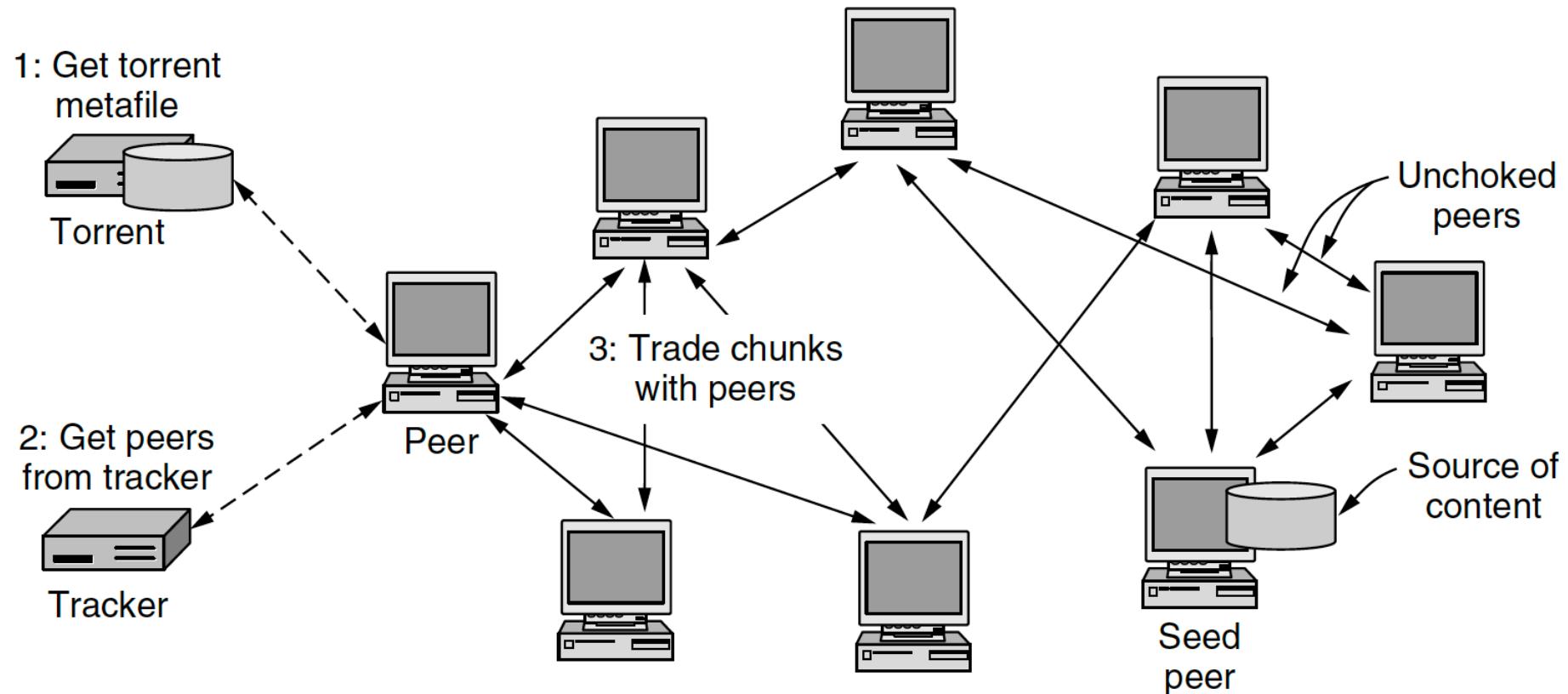
# 对等网络 (P2P, PEER-TO-PEER NETWORKS)

- 关键组成：
  - Tracker：一台用于维护所有正在上传、下载的peer的服务器
  - Swarm：这一群peer节点称为swarm
    - Swarm成员需要定期向tracker汇报自身是否处于激活状态，离开时同样需要报告。
    - 当有新peer加入swarm时，需要先与tracker通信以获知swarm中的其他peer节点。



# 对等网络 (P2P, PEER-TO-PEER NETWORKS)

*Getting the torrent and contacting the tracker are the first two steps for downloading content*



# 对等网络 (P2P, PEER-TO-PEER NETWORKS)

- 针对第二个问题 (本质为如何为用户提供快速下载) :
  - Seeders (种子节点) : 拥有内容的peer节点
  - Peer节点加入swarm的同时就会开始下载自身所缺失的内容, 同时将自身所拥有的内容上传给其他peer节点
    - 通常peer节点下载完成后还会待在swarm中一段时间
    - 优先下载稀缺内容以提升下载速率 (副本越多, 越不容易出现bottleneck)
  - 因为peer节点随时可以加入或离开swarm, 因此swarm的流失率可能极大。



# 对等网络 (P2P, PEER-TO-PEER NETWORKS)

- 针对第三个问题 (本质为如何激励用户来为别人提供下载) :
- Free-riders: 只下载不上传的用户
- 奖励机制: 奖励有着良好上传行为的用户
  - 用户在上传的同时可以从与他通信的节点下载内容
  - 通过随机匹配的方式确保新加入的用户可以获得内容



# 对等网络 (P2P, PEER-TO-PEER NETWORKS)

- CDN为集中式
  - 集中式管理缺点：
    - 管理节点会成为瓶颈 (当规模不断扩大，存在单一节点失效的问题)
- 早期P2P为部分分布式 (完全分布式会降低效率)
  - 节点信息 (存于tracker中) 为集中式
  - 是否可以将该节点信息做成分布式?



# 对等网络 (P2P, PEER-TO-PEER NETWORKS)

- DHTs (distributed hash tables):
  - 传统P2P不在乎peer节点如何连接，称为非结构化P2P网络
  - DHTs则不然，因此称为结构化P2P网络
  - 如何实现peer节点信息分布式呢？



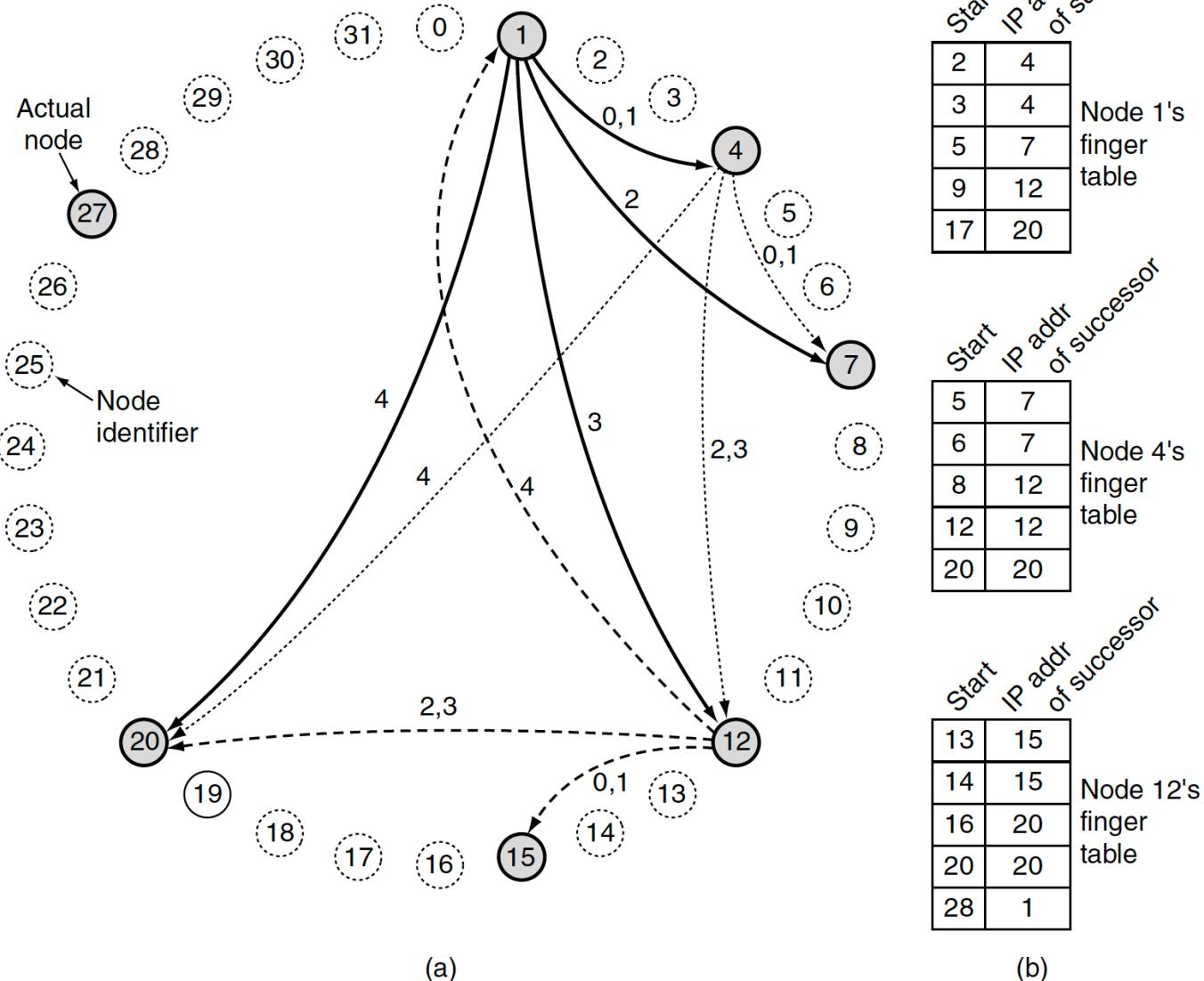
# 对等网络 (P2P, PEER-TO-PEER NETWORKS)

- Chord:

- 将节点组织成一个环状结构，并使用一致性哈希算法将键映射到环上的节点。每个节点负责一段哈希空间，并维护一张路由表，其中包含其他节点的信息。通过路由表，节点可以有效地查找存储和检索键-值对的位置



$n=2^m$ ,  $m = 5$ 的节点标识符环 (node identifier circle)



**Figure 7-71.** (a) A set of 32 node identifiers arranged in a circle. The shaded ones correspond to actual machines. The arcs show the fingers from nodes 1, 4, and 12. The labels on the arcs are the table indices. (b) Examples of the finger tables.

# 对等网络 (P2P, PEER-TO-PEER NETWORKS)

- Chord:
  - 使用ID来替代IP
    - 使用hash来将IP映射为m-bit的数字 (可排序)
      - 如使用SHA-1来将IP映射为160-bit的数字, 称为节点ID
  - 定义函数 $\text{successor}(k)$ , 表示在环形上顺时针绕过k后的第一个实际节点的节点标识符。例如,  $\text{successor}(6) = 7$ ,  $\text{successor}(8) = 12$ ,  $\text{successor}(22) = 27$ 。



# 对等网络 (P2P, PEER-TO-PEER NETWORKS)

- Chord: 潜在节点通过采用key-value结构来加入swarm
  - 注意: actual node为实际参与chord协议的节点, 其余节点均未实际参与协议
  - Key: 通过hash函数来将内容名称(name)转化为hash值, key=hash (torrent) , 此处name为torrent
  - Value: 节点的IP地址 (实际是hash得到的节点ID)
  - 找到successor(k)来存入节点ID
    - key-value:(hash(torrent), hash(IP))



# 对等网络 (P2P, PEER-TO-PEER NETWORKS)

- 如何根据key来查找value呢?
- 顺序查找
  - $n/2$  平均查找次数
- 其他方法?



# 对等网络 (P2P, PEER-TO-PEER NETWORKS)

- Finger table:
  - 有m个条目，从0至m-1编号
  - 每个条目由start跟后继节点ID构成
    - $start = k + 2^i \text{ (modulo } 2^m\text{), } i \in m$
    - 节点ID=successor ( $start[i]$ )

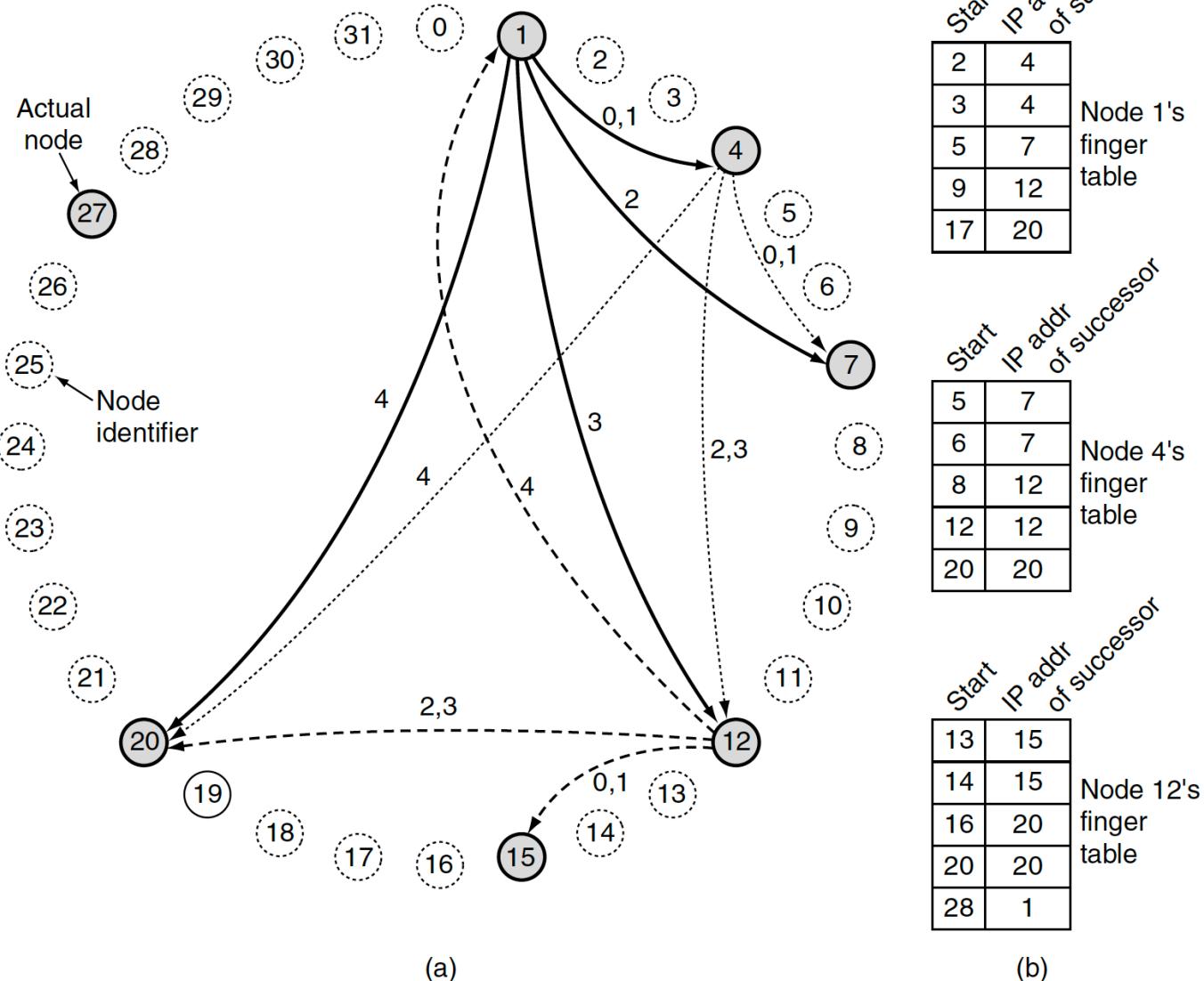


# 对等网络 (P2P, PEER-TO-PEER NETWORKS)

- 如何查表?
  - 1. key是否落在节点n与其后继节点之间, 是的话n的后继即为索要查找的节点
  - 2. 否则, 查找与key距离最近的后继, 并转入该后继节点
  - 3. 重复上述步骤
- Example:
  - 查找key=3
    - 4为所找节点
  - 查找key=16
    - 1->12->15>16



$n=2^m$ ,  $m = 5$ 的节点标识符环 (node identifier circle)



**Figure 7-71.** (a) A set of 32 node identifiers arranged in a circle. The shaded ones correspond to actual machines. The arcs show the fingers from nodes 1, 4, and 12. The labels on the arcs are the table indices. (b) Examples of the finger tables.

# 对等网络 (P2P, PEER-TO-PEER NETWORKS)

- Actual node  $r$ 如何加入?

- 1. 通过任意节点获知successor ( $r$ )
- 2. 通过successor ( $r$ ) 获知其前继 (predecessor)
- 3. successor ( $r$ ) 将节点 $r$ 作为前predecessor, 原predecessor将 $r$ 作为其新successor
  - E.g., 24想要加入
  - Finger table周期性的更新, 遇到新的successor即更新



# 对等网络 (P2P, PEER-TO-PEER NETWORKS)

- Actual node 离开时：
  - 通知其前继与后继
  - 如果发生意外、故障时怎么处理？
    - 每个节点不仅与其直接前继后继联系，还与其 $s-1$ 个连续的前继后继联系。



**Thank You**

