第五章 网络层（4）

袁华：hyuan@scut.edu.cn

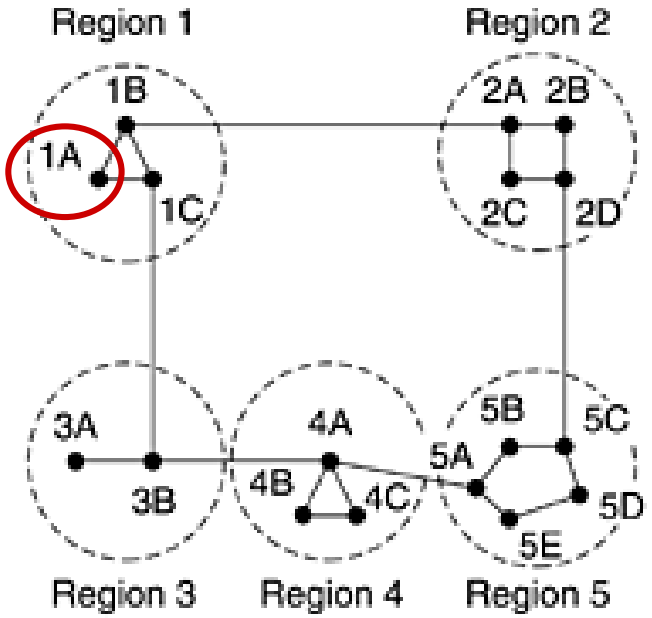
广东省计算机网络重点实验室

计算机科学与工程学院

本节主要内容（5.2.6~5.2.11 P292~302）

- 分级/层次路由（Hierarchical routing）
- 广播路由（Broadcast routing）
- 组播路由（Multicast routing）
- 选播/任播路由（Anycast routing）
- 移动主机的路由（Mobile routing）
- 移动自组网路由（Ad hoc routing）
- 对等网络节点查询（P2P）

分级路由 P292



(a)

Full table for 1A

Dest.	Line	Hops
1A	–	–
1B	1B	1
1C	1C	1
2A	1B	2
2B	1B	3
2C	1B	3
2D	1B	4
3A	1C	3
3B	1C	2
4A	1C	3
4B	1C	4
4C	1C	4
5A	1C	4
5B	1C	5
5C	1B	5
5D	1C	6
5E	1C	5

(b)

Hierarchical table for 1A

Dest.	Line	Hops
1A	–	–
1B	1B	1
1C	1C	1
2	1B	2
3	1C	2
4	1C	3
5	1C	4

(c)

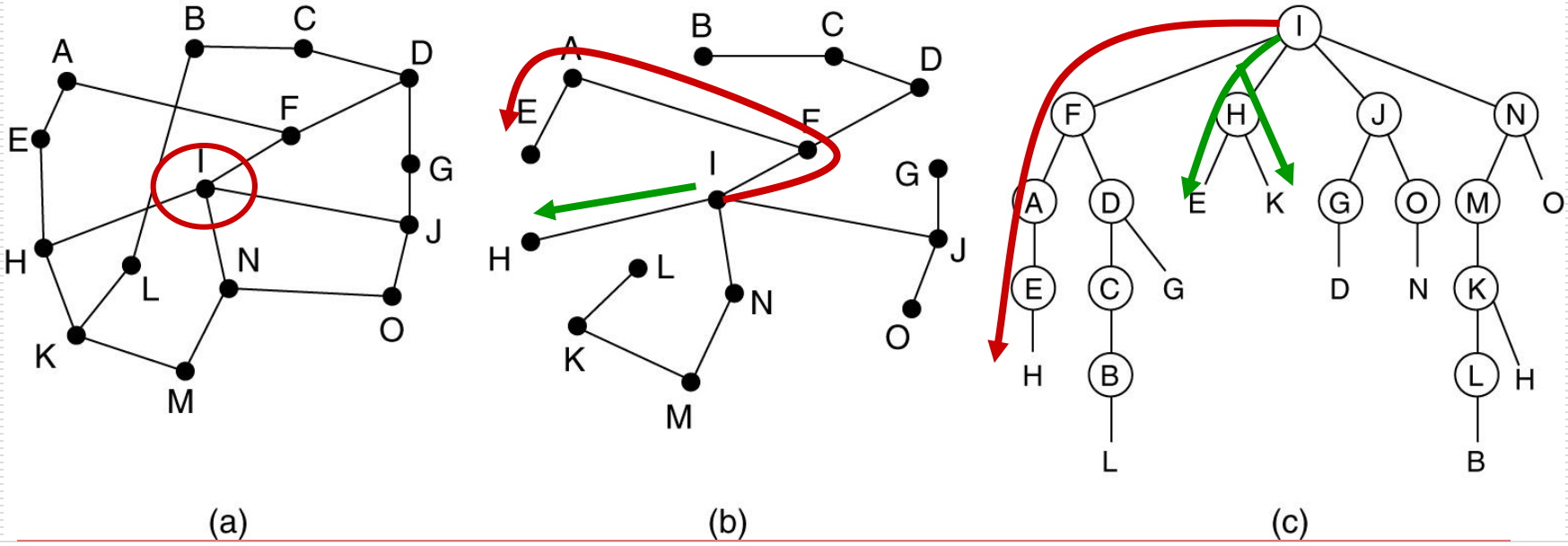
减少路由表规模!

广播路由 P293

- 可能的应用：
  - 天气预报发布、股票行情更新、现场直播节目等
- 广播路由实现的5种可能的方法：
  - 给每个目标单播每一个分组
  - 扩散法（Flooding）
  - 使用多目标路由（ multi-destination routing）
  - 使用汇集树/生成树（sink tree / spanning tree) 来引导分发分组
  - 使用逆向路径转发来控制扩散（flood）

逆向路径转发 P294

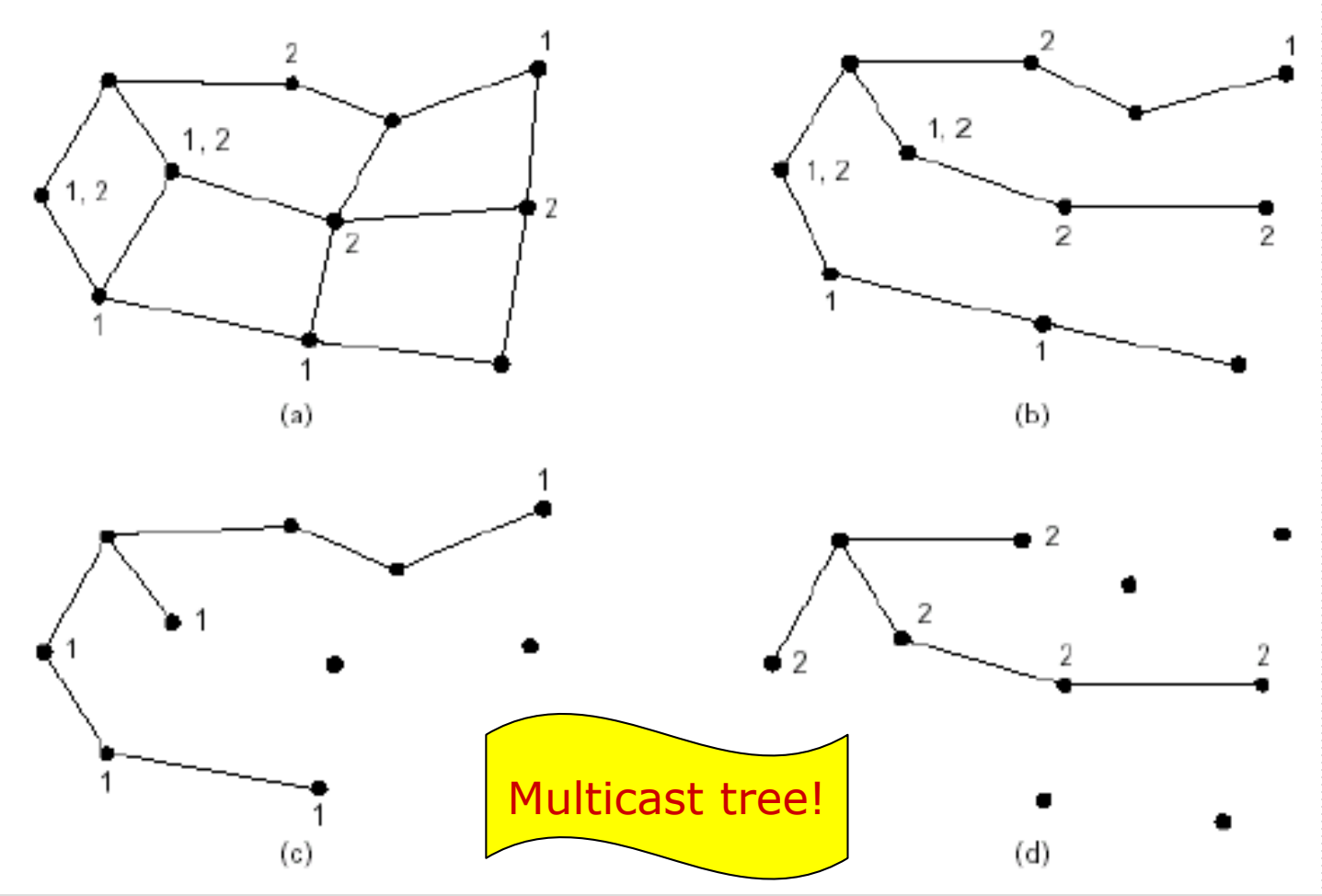
- 基本思想：当一个广播分组到达某个路由器的时候，如果它是从该路由器到广播源的通常线路上到达的，那么它被分发到所有的出口（除了来的那个口），否则被丢弃。



多播/组播路由（multicast）P295~296

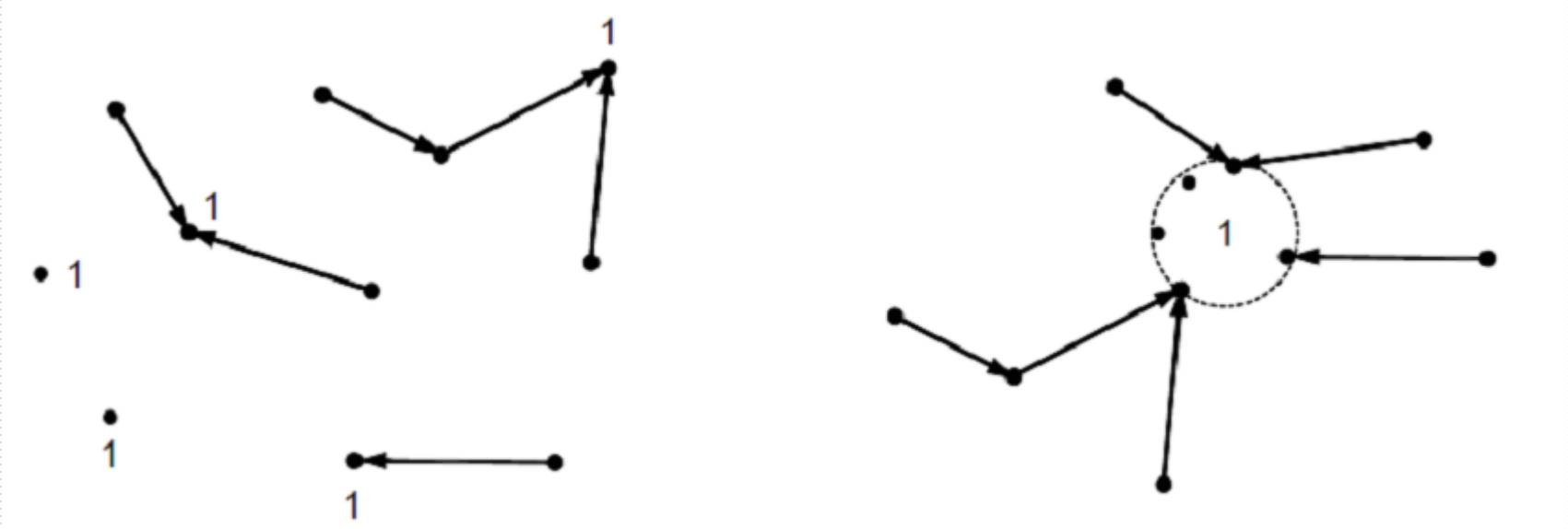
- IP支持组播，使用 D 类地址
- 每个 D 类地址标识了一组主机
  - 可以有 28 地址用来表示组，所有 2^{28}个组(224~239)
- IP组播的重要组成：
  - 成员管理 (IGMP/MLD)
  - 组播路由表 (DM/SM)
- IP组播必须要有特别的组播路由器的参与才能实现
  - 应用层组播（Application-layer multicast）

组播路由 P295



选播/任播路由P297

- 目的是一组节点，只需要发送到最近的那个。
- 典型应用：**DNS**

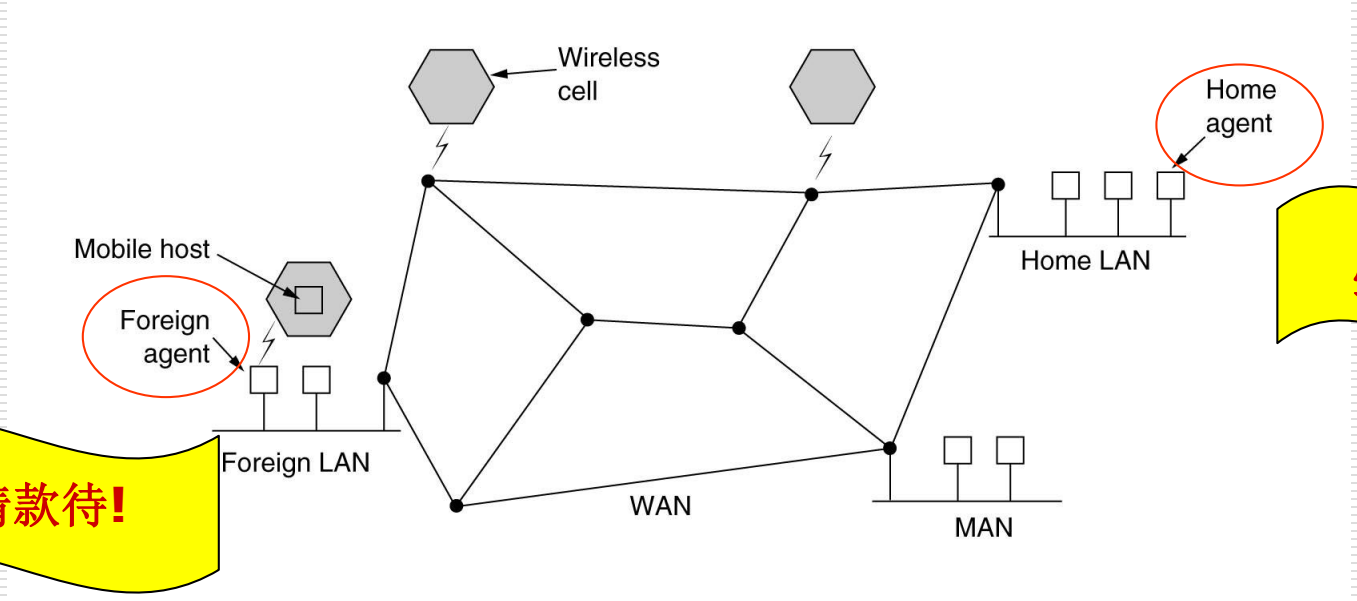


移动主机路由 P298

- 什么是移动主机？包括两类：
  - 迁移主机（**migratory host**），从一个地方移动到另一个地方，但是物理连上网络才能上网（如笔记本电脑）
  - 漫游主机（**roaming hosts** ）在移动中计算，当他们移动的时候，仍然保持与网络的连接
- 所有的主机都有一个永久的主场所（**home location**）和主地址（**home address**），主地址可用来确定它的主场所
- 包含移动主机的系统中，路由的目标就是：能够利用移动主机的主地址给它们发送分组，而不管它们实际在哪里；且要有效送达

移动主机的路由 （续P298~299）

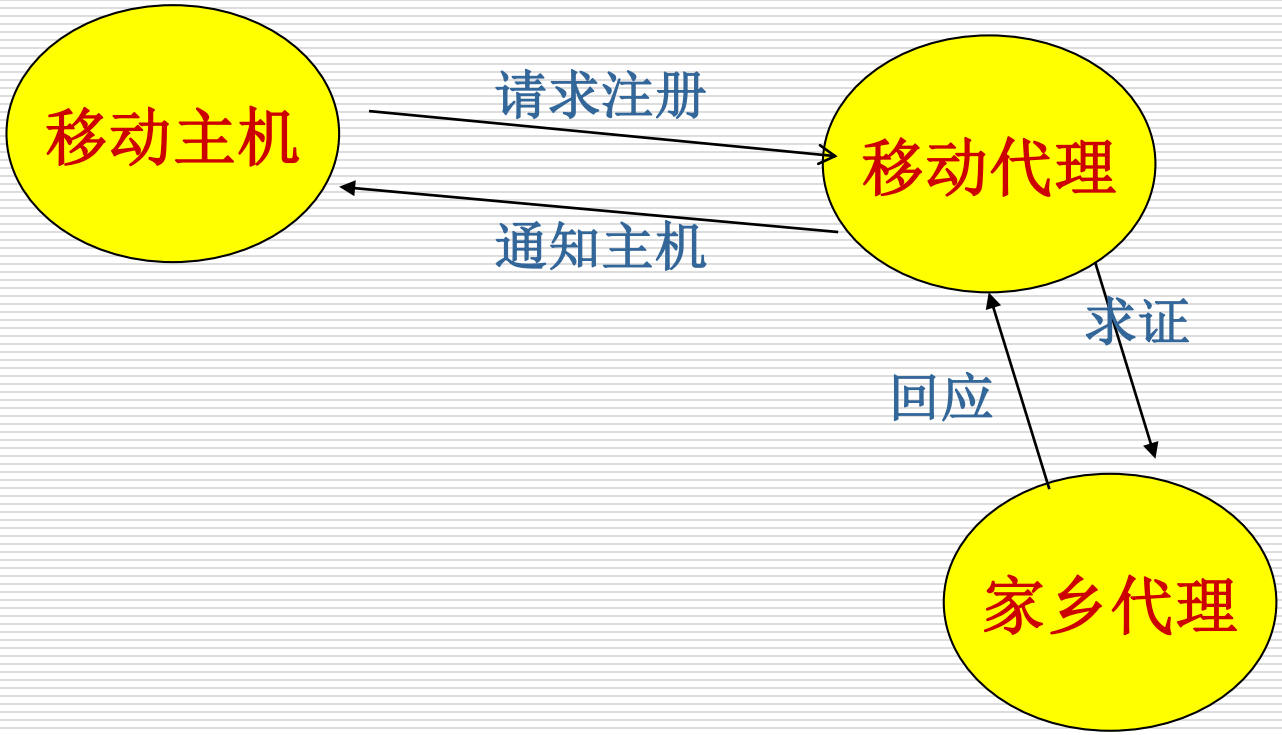
- 世界被分成若干小区域
  - 每个区域有一个或多个外部代理（**foreign agents**）， 它/它们记录下所有正访问该区域的移动主机
  - 每个区域有一个家乡/本地代理（**home agent**），记录下那些主场所在该区域，但是当前正在访问其他区域的主机



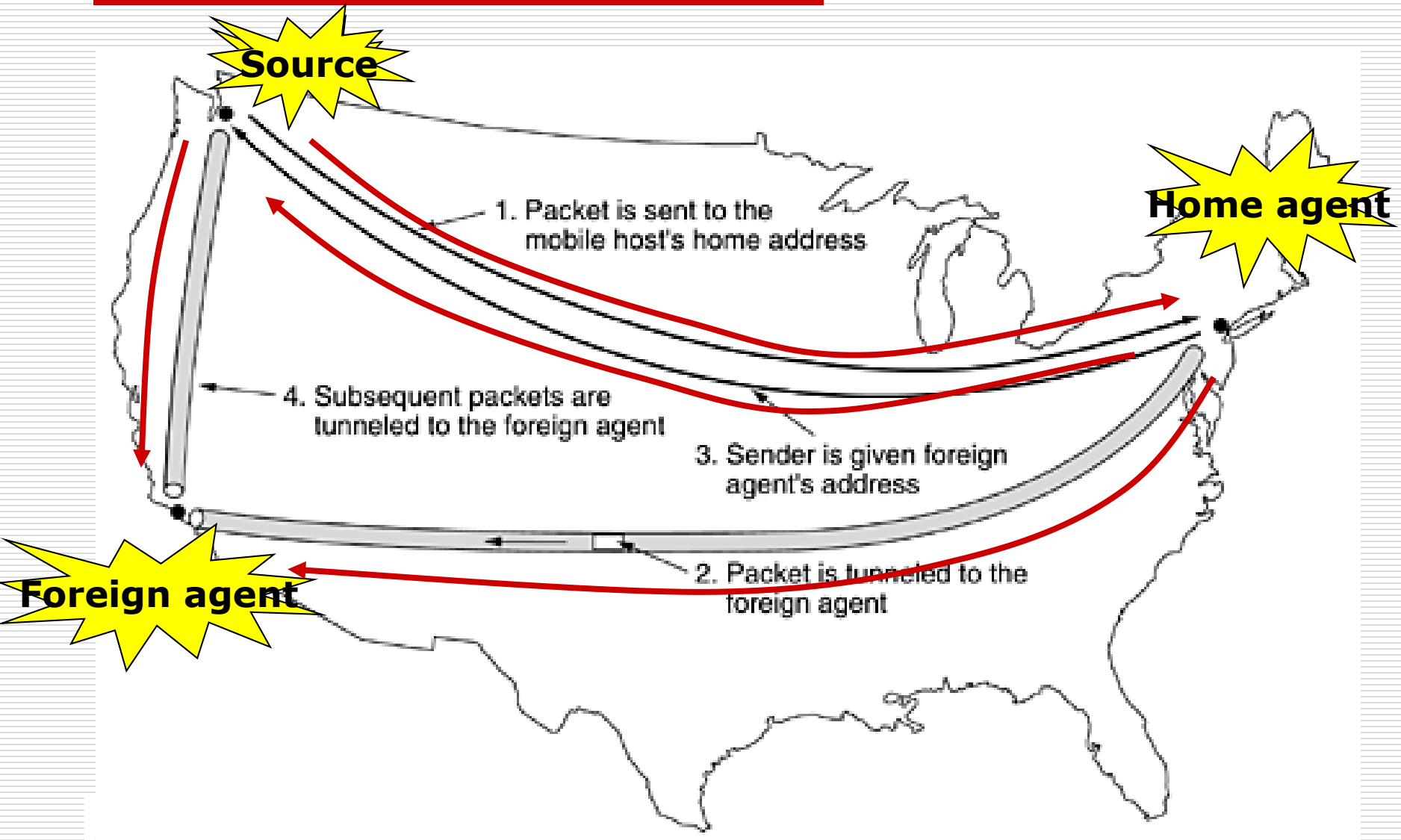
移动主机注册过程

- 每个外部代理（**foreign agent**）周期性地广播一个分组，宣布它的存在和地址
- 移动主机（**mobile host**）向外部代理请求注册，提供它的主/家乡地址，当前的数据链路层地址，以及一些安全信息
- 外部代理（**foreign agent**） 与移动主机的本地/家乡代理联系，告诉它，你的一个主机正在这里
- 家乡代理（**home agent**）检查外部代理提供的主机安全信息，如果一切都正确，则通知外部代理可以继续
- 当外部代理得到家乡代理的确认，它在本地表上增加一个表项，并通知移动主机，注册完成（**registered**）
- 当一个移动主机离开区域时，它应该宣布它的离开（ **de-registrate** ）

注册过程图示



移动主机的分组转发过程 (tunnel) P299



Ad Hoc 网络路由 P300

- Ad hoc网络 (or MANETs, Mobile Ad hoc NETworks) 中，主机和路由器都在移动
  - 战场上的军事设备 - No infrastructure
  - 大海上航行的船只
  - 地震中，紧急工作设施
  - 一群有笔记本电脑的人，但是没有 802.11.
- 在 ad hoc 网络中，拓扑一直在改变，它的路由和固定网络中的路由截然不同

P2P网络（Peer-to-Peer）P578

- 对等网络
- P2P网依赖于参与者（peer）的计算能力和带宽，而不是将控制力集中在少数的几个服务器
- P2P网络是分布式的，所有的节点都是对称的，没有中心控制，也没有层次组织
- 其应用完全不同于C/S和B/S

P2P网络（续）

- 优点
  - 所有的参与节点都提供资源，包括带宽、存储空间和计算能力
  - P2P网络的分布特性增加了健壮性（robustness）
- 非结构化（Unstructured）和结构化的 P2P 网络
  - 非结构化的 P2P 网络
    - 为了找到所需的数据，查询请求必须在网络上进行扩散，已发现尽可能多的拥有这些数据的peer
    - 很多著名的P2P网络都采用了非结构化的网络，如 Napster, Gnutella 和 KaZaA
  - 结构化的 P2P 网络
    - 维护一个分布式的哈希表（Distributed Hash Table ,DHT），每个peer负责网络上一部分的内容
    - 一些知名的结构化P2P网络如： Chord, Pastry, Tapestry, CAN and Tulip

P2P技术的应用

- 文件内容共享和下载，例如Napster、Gnutella、eDonkey、eMule、Maze、BT等；
- 计算能力和存储共享，例如SETI@home、Avaki、Popular Power等；
- 基于P2P技术的协同与服务共享平台，例如JXTA、Magi、Groove等；
- 即时通讯工具，包括ICQ、QQ、Yahoo Messenger、MSN Messenger等；
- P2P通讯与信息共享，例如Skype、Crowds、Onion Routing等；
- 基于P2P技术的网络电视：沸点、PPStream、 PPLive、QQLive、 SopCast、BanaCast等。

P2P 文件共享

Example

- Alice 在她的笔记本电脑中运行 P2P 客户程序
- 间歇地连接到网络，获取每个连接的新IP地址
- 查找 “Hey Jude”
- 应用程序将显示含有 “hey jude”信息的所有 peer
- Alice选择其中的一个peer，Bob.
- 文件从Bob的 PC 拷贝到 Alice的笔记本电脑
- 当 Alice下载的同时，别的用户可以从Alice下载
- Alice的peer是一个web客户端，同时也是一个临时的web 服务器

peers are servers = highly scalable!

P2P: 集中目录

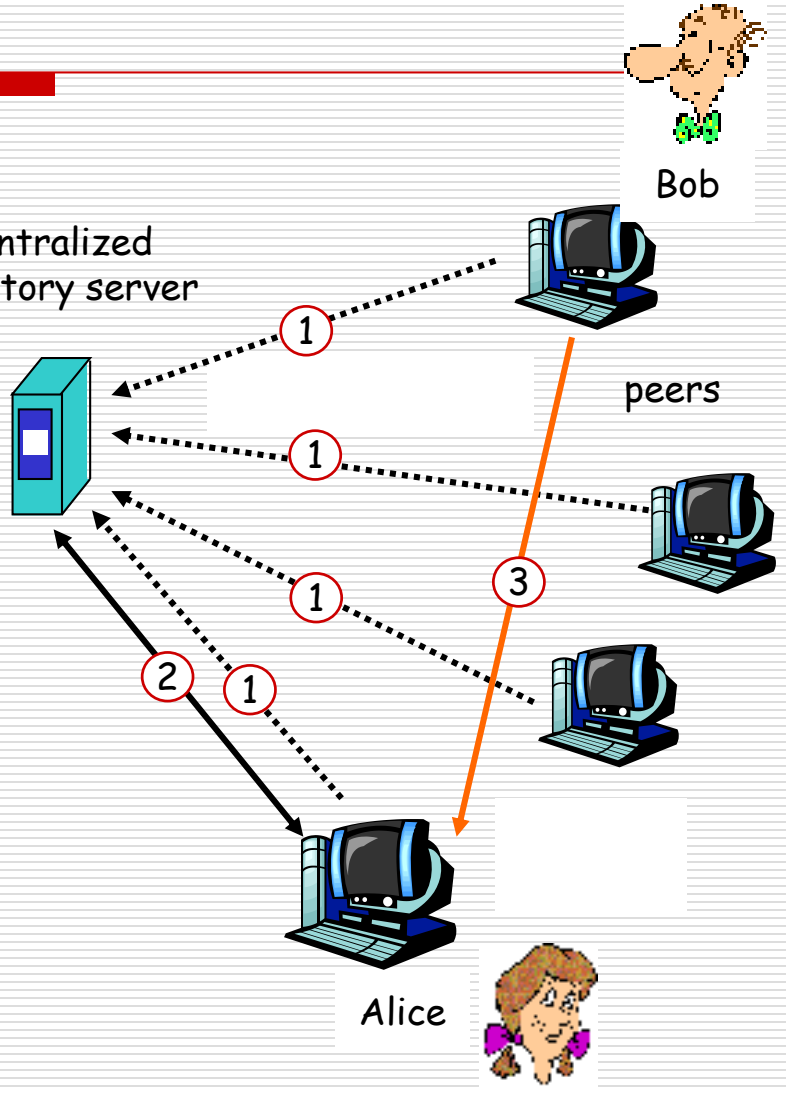
Original "Napster" design

1)当 peer 连接上网的时候，它通知中心服务器：

- **IP地址**
- **内容**

2) Alice向中心服务器请求"Hey Jude"

3) Alice 向Bob请求



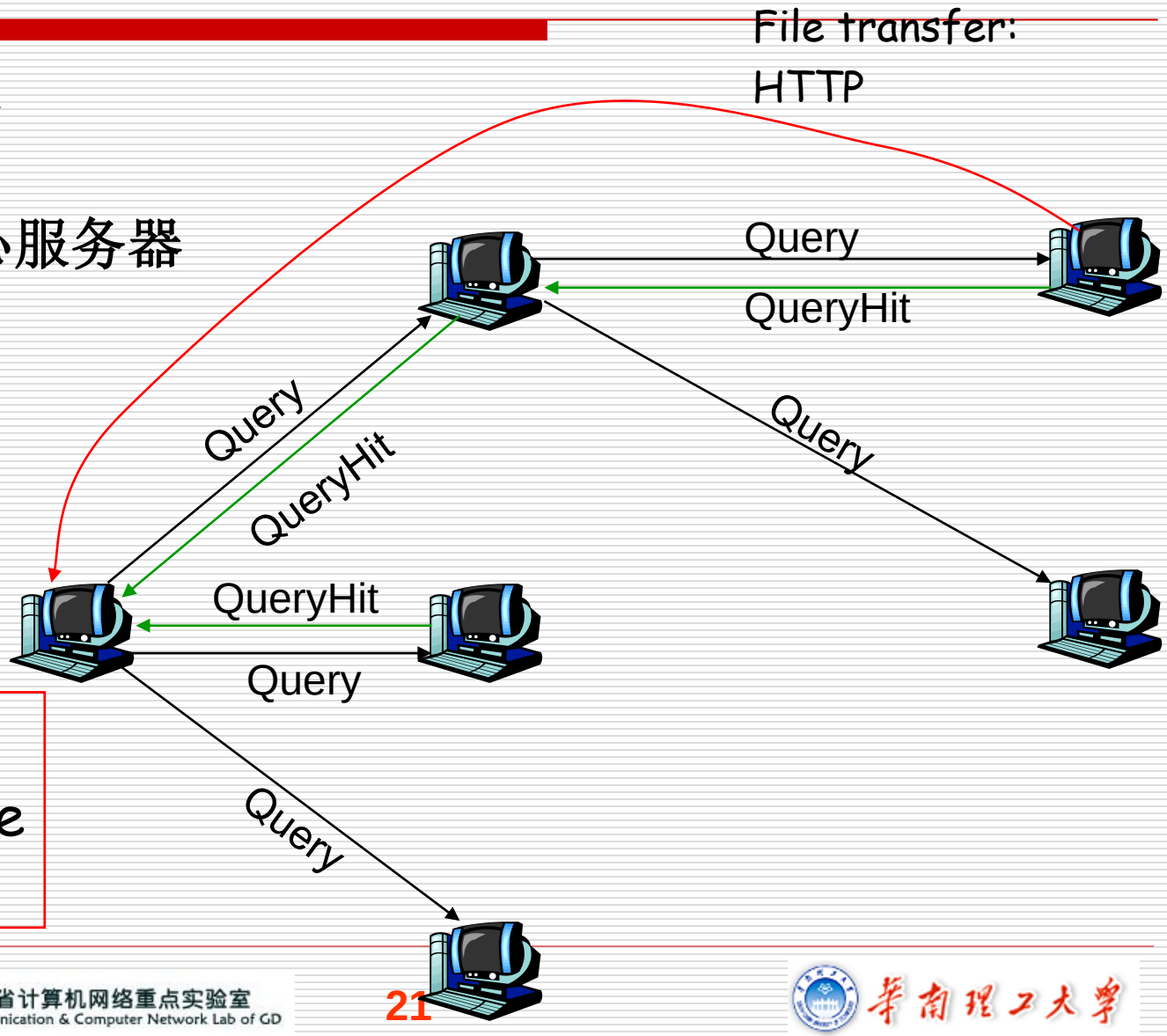
P2P: 集中目录式的问题

- 可能单点故障
- 性能瓶颈
- 版权问题

文件传输是分布式的，但是信息/内容定位是高度集中的

Gnutella：协议

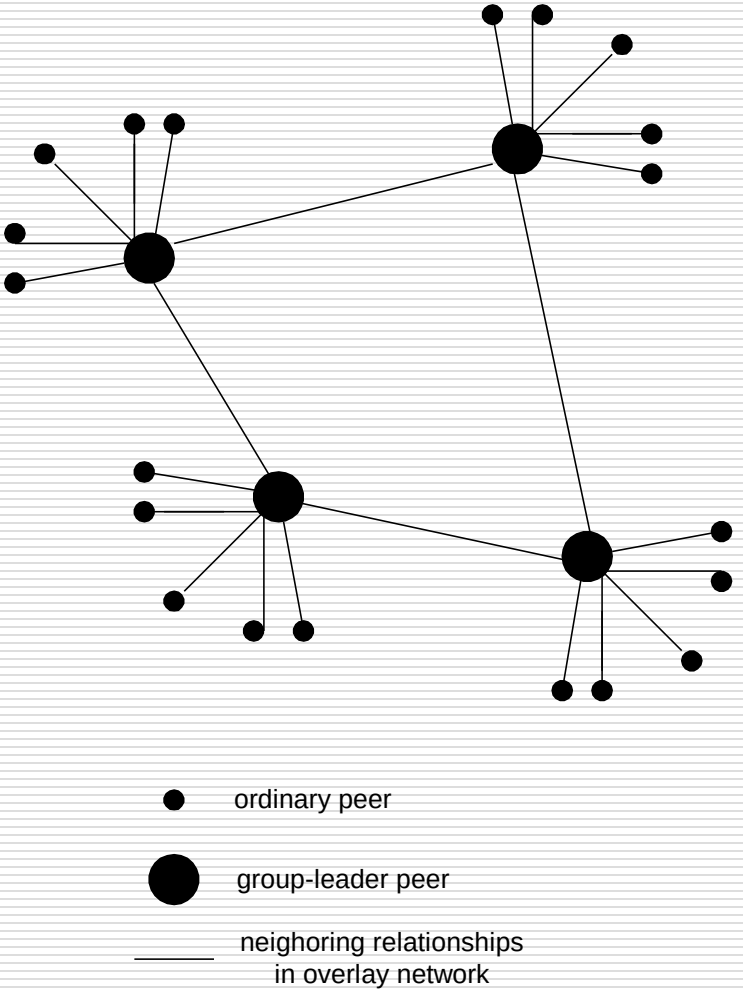
- 全分布
  - 没有中心服务器



Scalability:
limited scope
flooding

发掘异构性：KaZaA

- 一个peer 要么是一个群领袖，要么就是属于一个群领袖（group leader）
  - Peer和他的群领袖之间采用TCP连接
  - 一些群领袖之间采用TCP连接
- 群领袖记录下他所有的属下peer的内容



KaZaA: 查询

- 每个文件有个hash值和描述descriptor
- 客户向他的群领袖发送查询
- 回应
  - 群领袖如果找到匹配项，回应：
    - 匹配项： metadata, hash, IP address
  - 群领袖将查询向其他群领袖转发
- 客户从回应中选取一个来下载

Chord 中的节点查询 P584

- Chord由麻省理工学院（MIT）在2001年提出，是一种DHT(分布式哈希表)实现
- 每个用户节点有 IP 地址，使用哈希函数生成 m 位的哈希值
  - Chord 使用 SHA-1生成hash.
  - IP地址被映射为160位散列值，称为 节点标识符（node identifier ）
- 概念上讲，所有 2^{160} 个节点标识符，按升序排列组成一个大圆 big circle.
- 圆上的一些节点标识符对应的节点是参加网络的节点，但大部分的节点标识符不是。

Chord 中的节点查询(续)

- 定义函数 successor(k)，它的返回值是：从圆上顺时针方向遇到的第一个实际节点的节点标识符
- 记录的名字，也使用 hash函数 (如 SHA-1) 生成160位的散列值，称为键（ key)
  - key = hash(name)
- 为了让其他人可以访问这条记录，拥有这条记录的节点建立起一对关联 (name, my-IP-address)，然后让 successor(hash(name)) 来存储这对关联
- 采用这种方式，内容/记录的索引信息被随机地分布在节点上
- 为了容错，可以使用 p 个不同的 hash 函数来计算，这样同一条记录的索引信息被存储在 p 个节点上

Chord 中的节点查询(续)P584

- 如果用户想查找 **name**记录,他首先**hash**计算得到 **key** 值，然后使用 **successor (key)** 得到存放索引关联信息的实际节点的**IP**地址
- 查询过程：
  - 请求节点发送分组给它的后继节点，分组中包含它自己的**IP**地址和待查找记录的 键 值
  - 分组沿着环向前传播，直到到达被查询的节点标识符的后继节点
  - 节点对分组进行检查，看自己是否有合适的信息匹配键值，如果有，将相关的信息返回请求节点

Chord 中的节点查询优化P585

- 第一个优化措施：每个节点同时保留前继节点和后继节点，所以查询可以沿着顺时针方向进行，也可以沿着逆时针方向进行
- 在一个巨大的P2P系统线性搜索节点，效率很低，每次搜索涉及到的节点平均为 n/2
- 为了加快搜索，每个节点维护着一个指取表finger table
  - 指取表有 m 个表项，索引值从 0 到 m – 1，每个表项指向一个不同的实际节点
  - 每个表项有两个域：
    - start
    - successor(start)的IP地址
  - 在节点K上第i个表项的两个域分别为：

$$start = k + 2^i \quad (modulo\, 2^m)$$

IP address of $successor(start[i])$

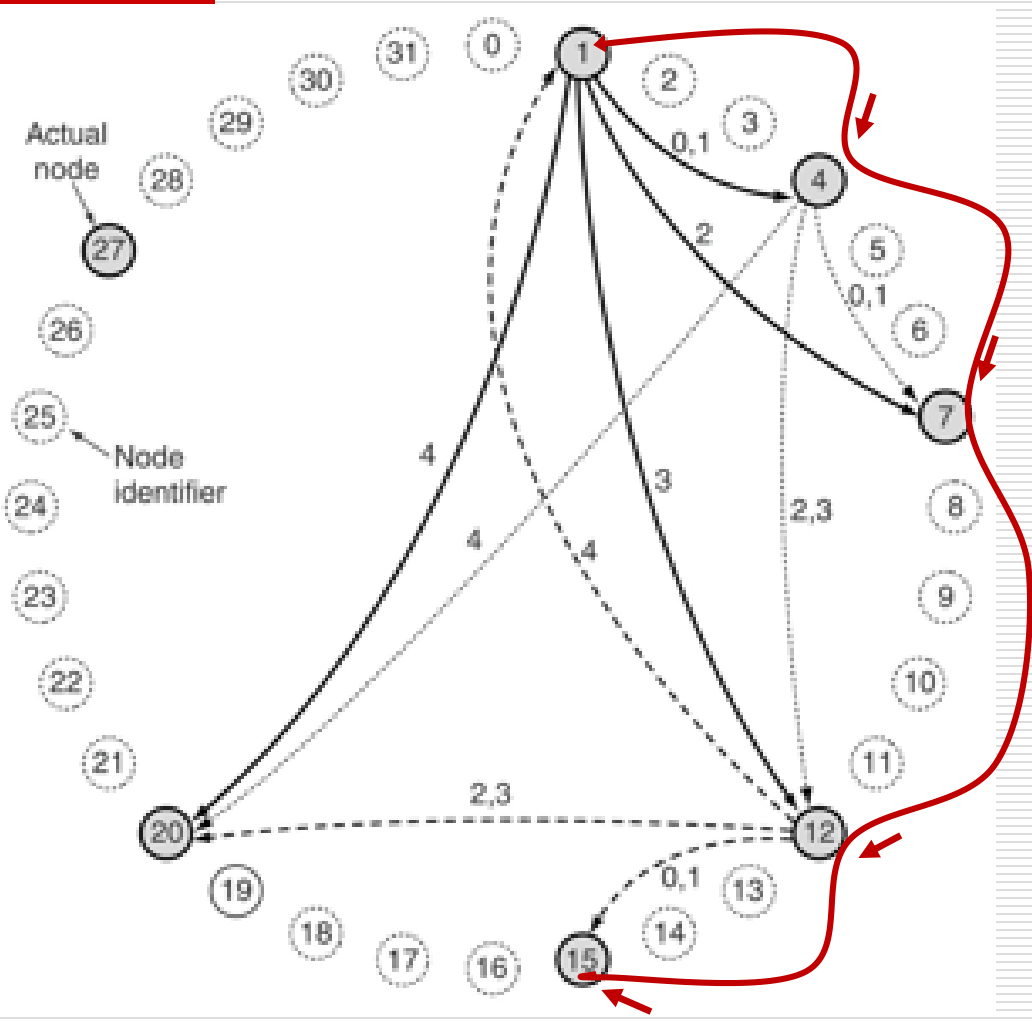
Chord 中的节点查询优化P585

- 使用了指取表之后，在节点 k 上查找 key值的过程：
  - 如果key值落在k 和后继节点 (k)之间，那么记录了带查询信息的节点就是 successor (k)，搜索成功，结束
  - 否则，节点k查找它的指取表，看哪个表项的start域在key的前面且最靠近key； 然后请求直接发送给该表项指示的IP地址，继续搜索
  - 平均查找次数为：$\log_2 n$

M=5的Chord圆

- 在节点1上查找key=3
- 在节点1上查找key=14
- 在节点1上查找key=16

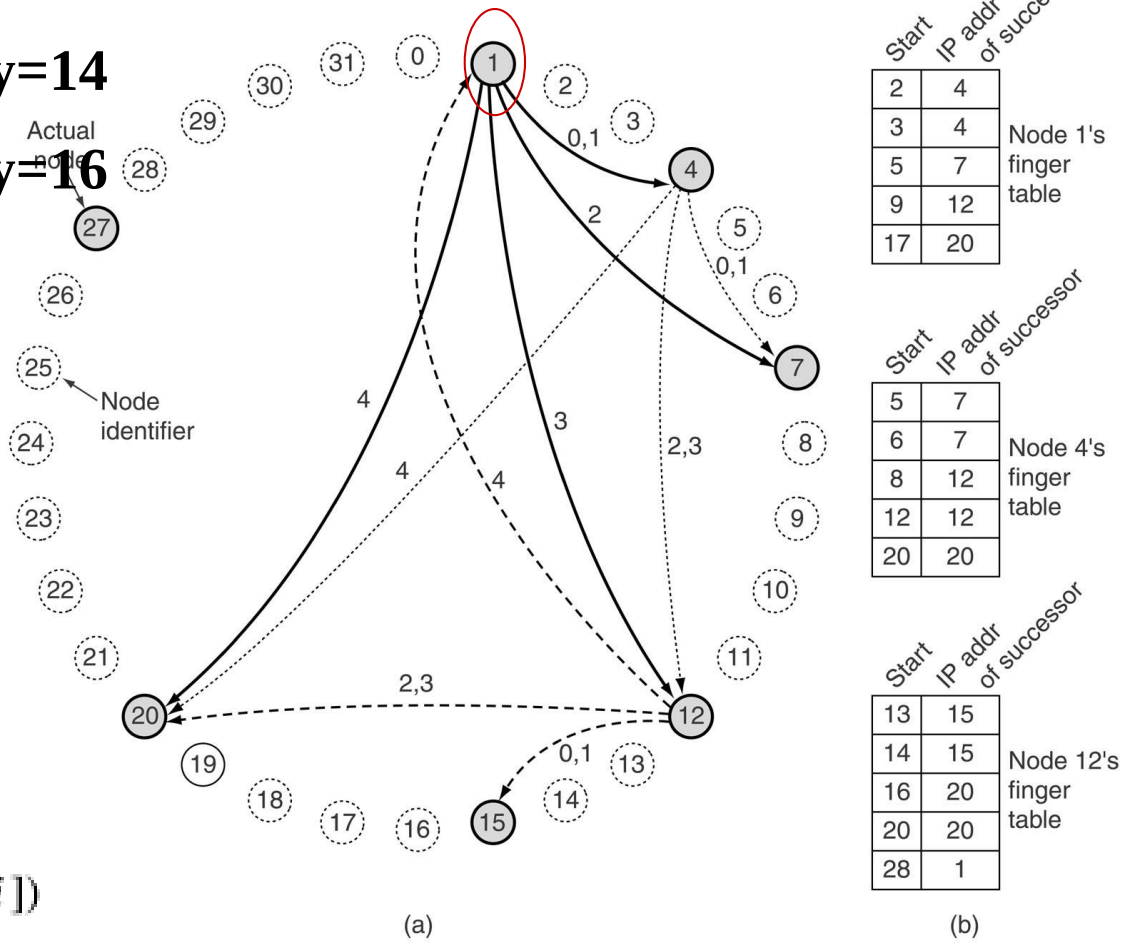
Key是什么？



Chord节点查找实例P584

- 在节点1上查找key=3
- 在节点1上查找key=14
- 在节点1上查找key=16

K=1，i=0,1,2,3,4



$$start = k + 2^{i} \quad (modulo\, 2^{m})$$

IP address of $successor(start[i])$

节点的加入和离开P585

- 当一个新节点 r 想加入网络，它必须和一个已有的节点进行联络，并请他为自己查找 successor (r) 的IP地址
- 新节点请 successor (r) 查找它的前继节点 predecessor.
- 新节点请求两者将自己插在他们之间，这样r就在圆上了
- 当一个节点要正常离开网络的时候，通知它的前继节点，以便前继节点可以将它的后继节点指向待离开节点的后继节点
- 但是，如果一个节点是意外离开的呢？
  - 每个节点不仅要指向它的后继节点，也要和s步以内的其他后继保持联系

本节小结

- 分级路由（Hierarchical routing）
- 广播路由（Broadcast routing）
- 组播路由（Multicast routing）
- 选播路由
- 移动主机的路由（Mobile routing）
- 移动自组网路由（Ad hoc routing）
- 对等网络节点查询（P2P）

第一部分内容小结

- 一个分组如何从源到达目的？
- 路由算法
  - **DV: RIP**
  - **LS: OSPF**
  - **BGP**
  - 其它

Thank you all !