整理自哈工大计算机网络MOOC课件

4. 网络层

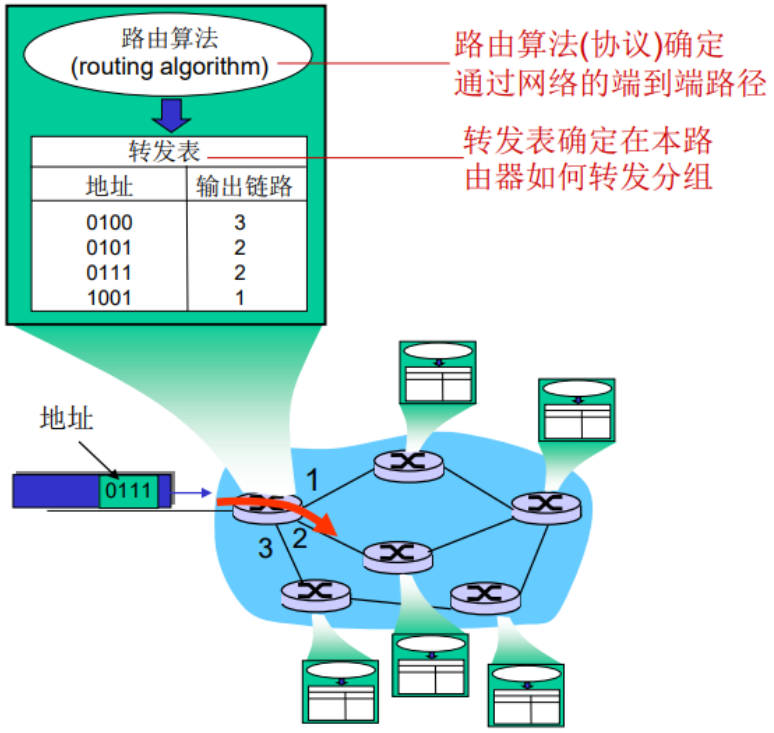
4.1 网络层服务

4.1.1 网络层

- 从发送主机向接收主机传送数据段（segment）
- 发送主机：将数据段封装到数据报（datagram）中
- 接收主机：向传输层交付数据段（segment）
- 每个主机和路由器都运行网络层协议
- 路由器检验所有穿越它的IP数据报的头部域
    - 决策如何处理IP数据报

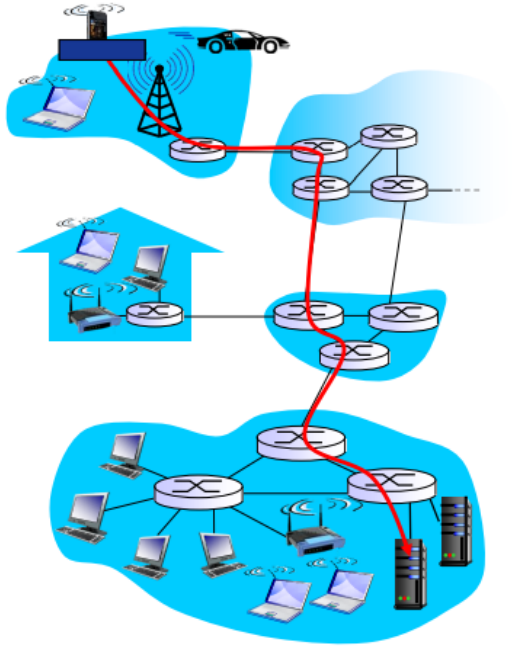
4.1.2 网络层核心功能-转发与路由

- 转发(forwarding): 将分组从路由器的输入端口转移到合适的输出端口
- 路由(routing): 确定分组从源到目的的经过的路径
    - 路由算法（routing algorithms）



4.1.3 网络层核心功能-连接建立

- 某些网络的重要功能:
  - ATM, 帧中继, X.25
- 数据分组传输之前两端主机需要首先建立虚拟/逻辑连接
  - 网络设备（如路由器）参与连接的建立
- 网络层连接与传输层连接的对比:
  - 网络层连接: 两个主机之间 (路径上的路由器等网络设备参与其中)
  - 传输层连接: 两个应用进程之间（对中间网络设备透明）



4.1.3 网络层服务模型

Q: 网络层为发送端（主机）到接收端（主机）的数据报传送“通道(channel)”提供什么样的服务模型(service model)?

Network Architecture	Service Model	Guarantees ?				Congestion feedback
		Bandwidth	Loss	Order	Timing	
Internet	best effort	none	no	no	no	no (inferred via loss)
ATM	CBR	constant rate	yes	yes	yes	no congestion
ATM	VBR	guaranteed rate	yes	yes	yes	no congestion
ATM	ABR	guaranteed minimum	no	yes	no	yes
ATM	UBR	none	no	yes	no	no

CBR：固定比特率

VBR：可变比特率

ABR：可用比特率

UBR：不保证比特率

4.1.4 网络层服务模型

- 无连接服务(connection-less service)
  - 不事先为系列分组的传输确定传输路径
  - 每个分组独立确定传输路径
  - 不同分组可能传输路径不同
  - 数据报网络(datagram network)

- 连接服务(connection service)
    - 首先为系列分组的传输确定从源到目的经过的路径 (建立连接)
    - 然后沿该路径（连接）传输系列分组
    - 系列分组传输路径相同
    - 传输结束后拆除连接
    - 虚电路网络(virtual-circuit network)

4.2 虚电路网络与数据报网络

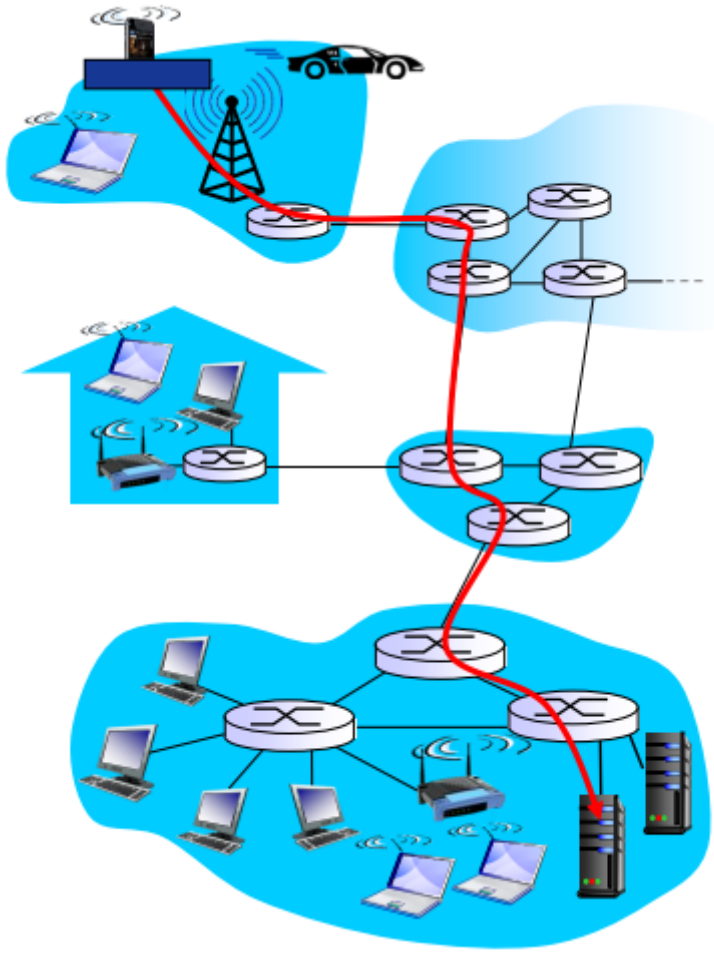
4.2.1 连接服务与无连接服务

- 数据报(datagram)网络与虚电路(virtual-circuit)网 络是典型两类分组交换网络
- 数据报网络提供网络层无连接服务
- 虚电路网络提供网络层连接服务
- 类似于传输层的无连接服务（UDP）和面向连接 服务（TCP），但是网络层服务是：
    - 主机到主机服务
    - 由网络核心实现

4.2.2 虚电路(Virtual circuits)

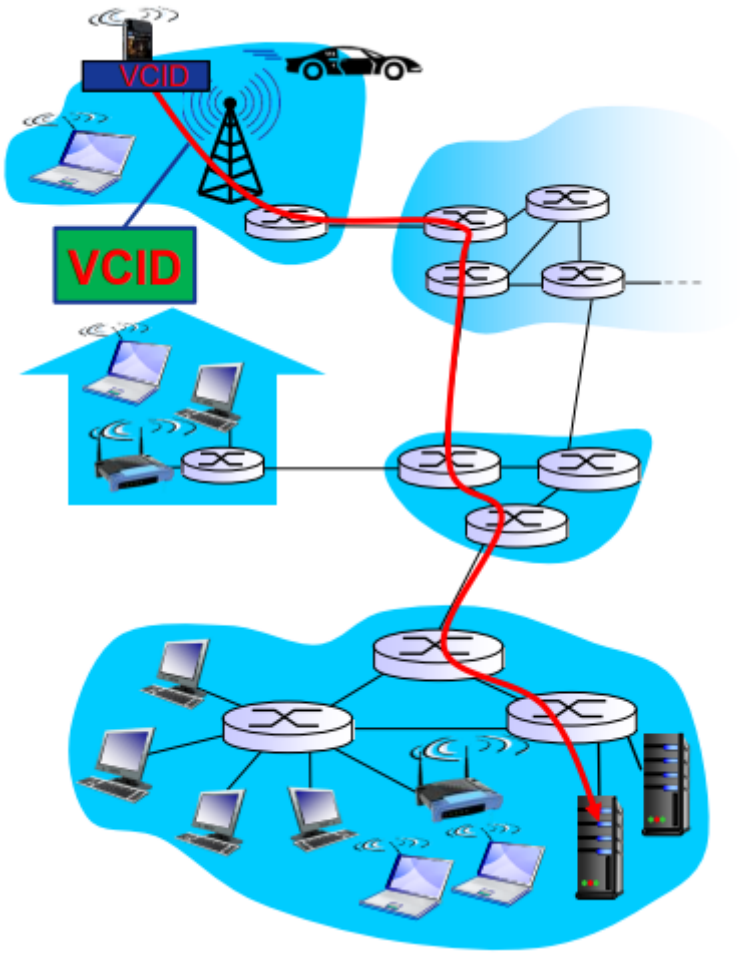
虚电路：一条从源主机到目的主机，类似于电路的路径(逻辑连接)

- 分组交换
- 每个分组的传输利用链路的全部带宽
- 源到目的路径经过的网络层设备共同完成虚电 路功能



- 通信过程

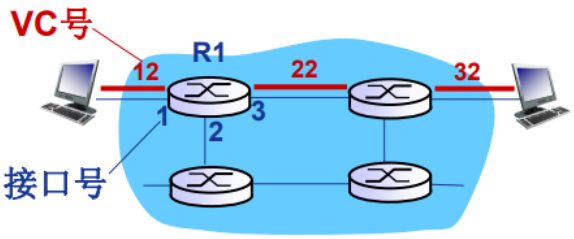
    - 呼叫建立(call setup)→数据传输 →拆除呼叫
- 每个分组携带虚电路标识(VCID)，而不是目的主机地址
- 虚电路经过的每个网络设备（如路由器），维护每条经过它的虚电路连接状态
- 链路、网络设备资源(如带宽 、缓存等)可以面向VC进行预分配
    - 预分配资源=可预期服务性能
    - 如ATM的电路仿真（CBR）



虚电路的具体实现

- 每条虚电路包括:
    1. 从源主机到目的主机的一条路径
    2. 虚电路号（VCID），沿路每段链路一个编号
    3. 沿路每个网络层设备（如路由器），利用转发表记录经过的每条虚电路
- 沿某条虚电路传输的分组，携带对应虚电路的 VCID，而不是目的地址
- 同一条VC，在每段链路上的VCID通常不同
    - 路由器转发分组时依据转发表改写/替换虚电路号

虚电路转发表



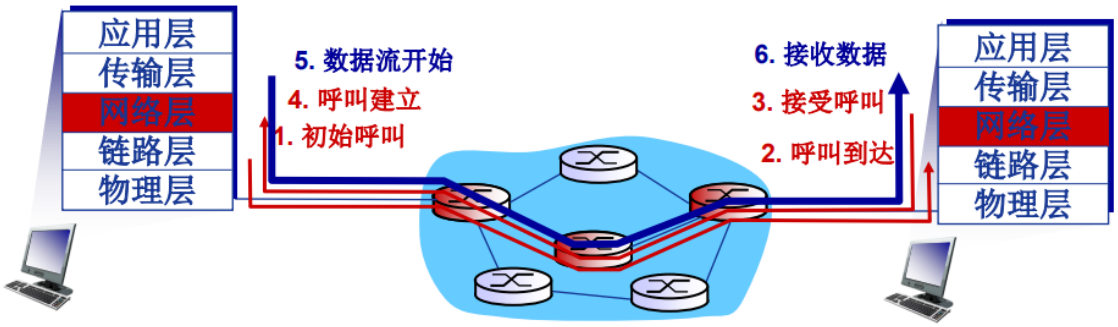
路由器R1的VC转发表:

输入接口	输入VC #	输出接口	输出VC #
1	12	3	22
2	63	1	18
3	7	2	17
1	97	3	87
…	…	…	…

VC路径上每个路由器都需要维护VC连接的状态信息！

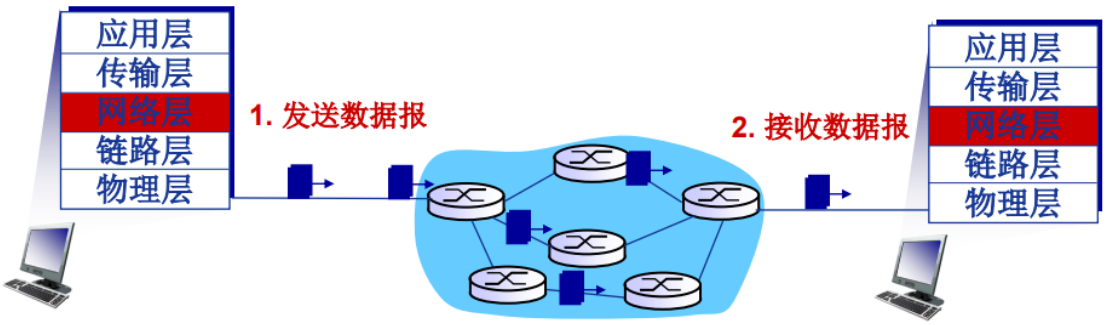
虚电路信令协议(signaling protocols)

- 用于VC的建立、维护与拆除
  - 路径选择
- 应用于虚电路网络
  - 如ATM、帧中继(frame-relay)网络等
- 目前的Internet不采用

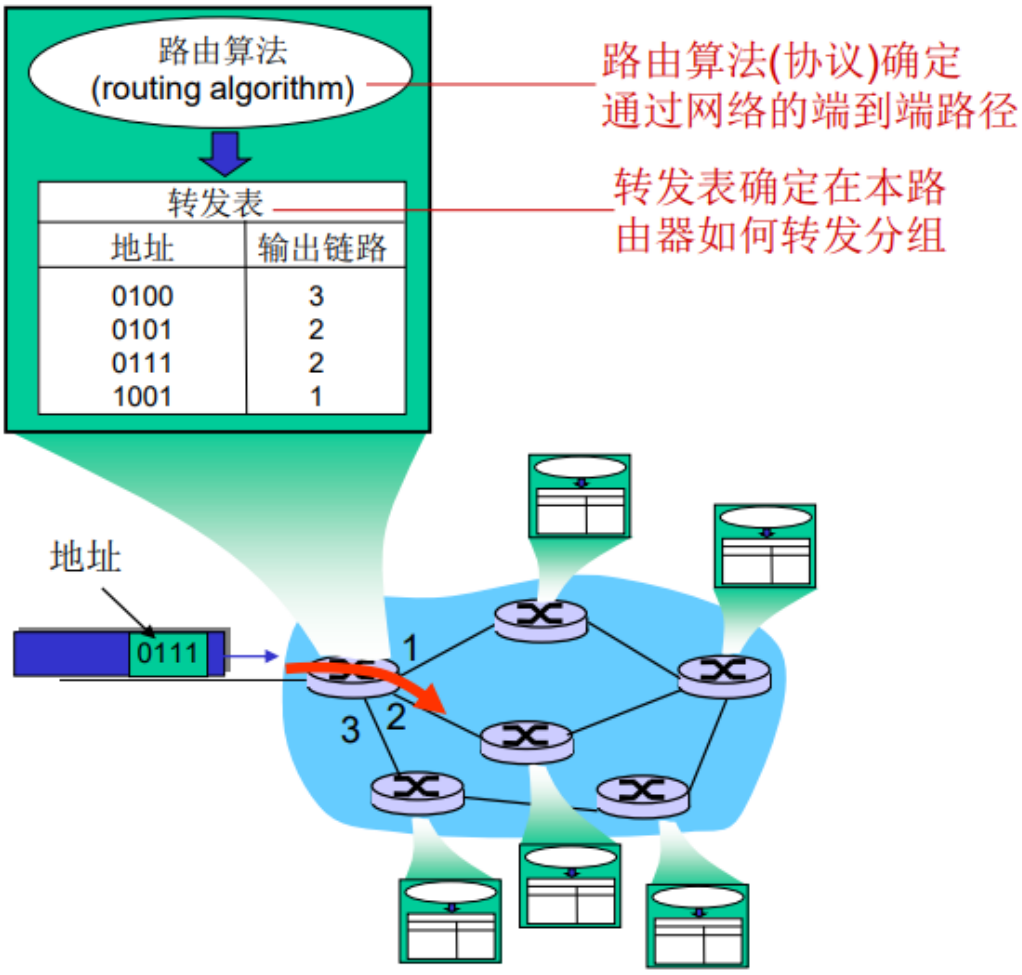


4.2.3 数据报网络

- 网络层无连接
- 每个分组携带目的地址
- 路由器根据分组的目的地址转发分组
  - 基于路由协议/算法构建转发表
  - 检索转发表
  - 每个分组独立选路



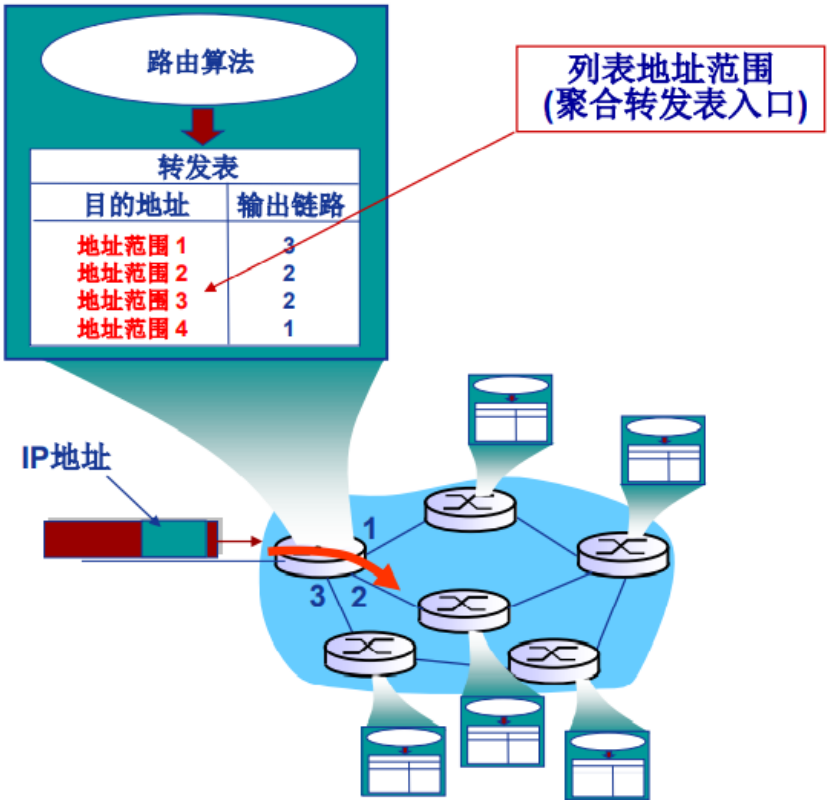
数据报转发表



IP地址太多，如果一一对应，则转发表太大

解决办法

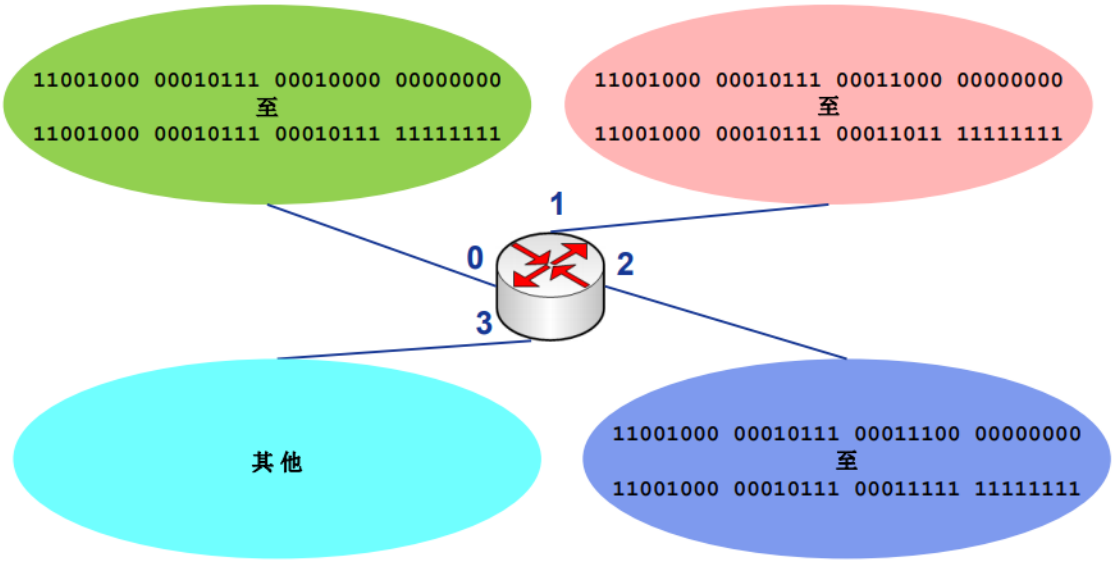
-



-

目的地址范围	链路接口
11001000 00010111 00010000 00000000 至 11001000 00010111 00010111 11111111	0
11001000 00010111 00011000 00000000 至 11001000 00010111 00011011 11111111	1
11001000 00010111 00011100 00000000 至 11001000 00010111 00011111 11111111	2
其他	3

-



Q: 如果地址范围划分的不是这么“完美”会怎么样？

例如：

目的地址范围	链路接口
11001000 00010111 00010*** *********	0
11001000 00010111 00011000 *********	1
11001000 00010111 00011*** *********	2
其他	3

- DA: 11001000 00010111 00010110 10100001 从哪个接口转发？ A:0
- DA: 11001000 00010111 00011000 10101010 从哪个接口转发？ A:1

最长前缀匹配优先

在检索转发表时，优先选择与分组目的地址匹配前缀最长的入口（entry）。

4.2.4 虚电路or数据报

Internet (数据报网络)

- 计算机之间的数据交换
  - “弹性”服务，没有严格时间需求
- 链路类型众多
  - 特点、性能各异
  - 统一服务困难
- “智能”端系统 (计算机)
  - 可以自适应、性能控制、差错恢复
- *简化网络，复杂“边缘”*

ATM (VC网络)

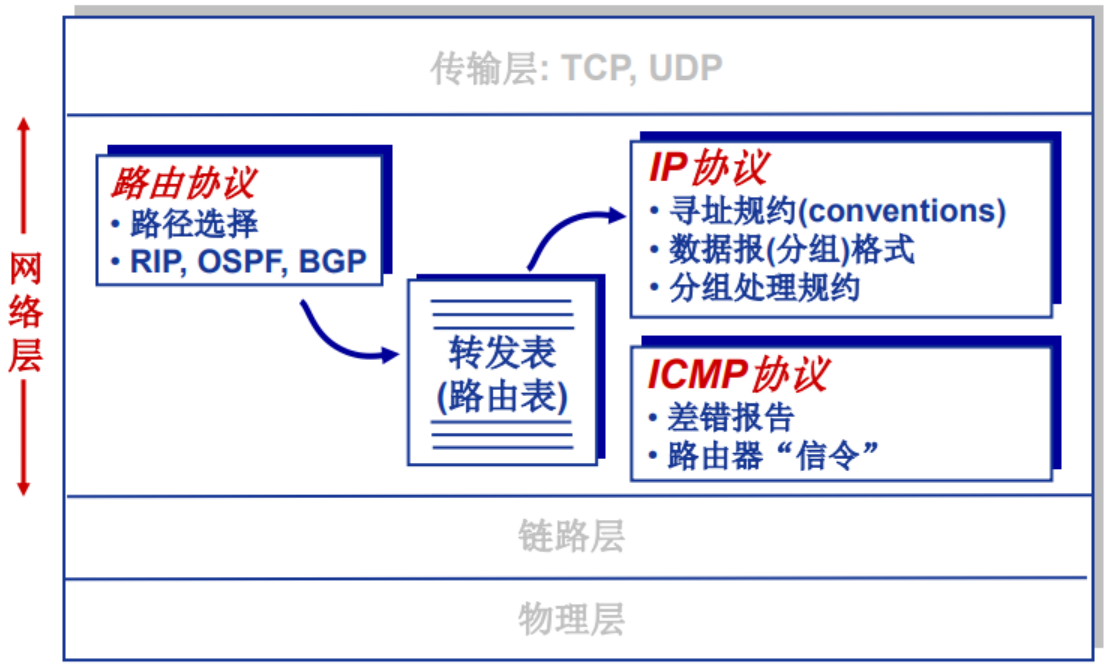
- 电话网络演化而来
- 核心业务是实时对话：
  - 严格的时间、可靠性需求
  - 需要有保障的服务
- “哑(dumb)” 端系统（非智能）
  - 电话机
  - 传真机
- *简化“边缘”，复杂网络*

4.3 IPv4协议

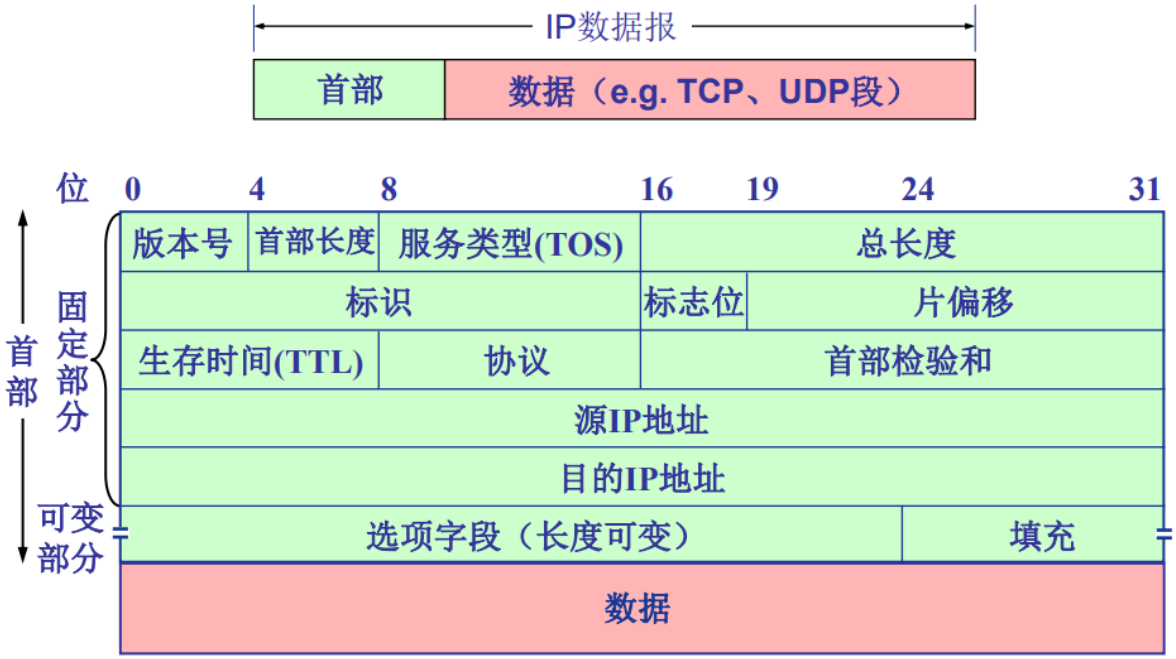
4.3.1 IP数据报

Internet网络层

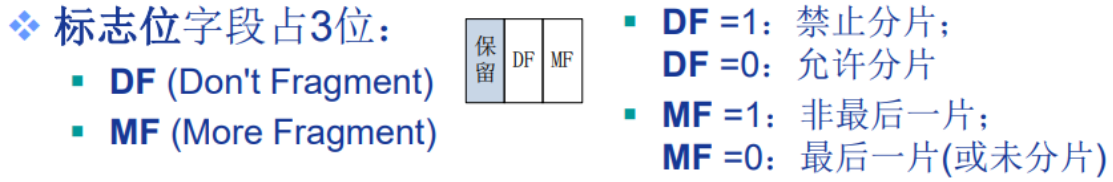
主机、路由器网络层主要功能：



IP数据报（分组）格式



- **版本号**字段占**4位**：IP协议的版本号
  - E.g：4→IPv4，6 → IPv6
- **首部长度**字段占**4位**：IP分组首部长度
  - 以4字节为单位
  - E.g：5→IP首部长度为20(5×4)字节
- **服务类型(TOS)字段**占**8位**：指示期望获得哪种类型的服务
  - 1998年这个字段改名为区分服务
  - 只有在网络提供区分服务(DiffServ)时使用
  - 一般情况下不使用，通常IP分组的该字段(第2字节)的值为00H
- **总长度**字段占**16位**：IP分组的总字节数(首部+数据)
  - 最大IP分组的总长度：65535B
  - 最小的IP分组首部：20B
  - IP分组可以封装的最大数据：65535-20=65515B
- **标识**字段占**16位**：标识一个IP分组
  - IP协议利用一个计数器，每产生IP分组计数器加1，作为该IP分组的标识，当数据报进行分片处理后，每个分片的标识值都与原数据报的标识值相同，则在接收端具有相同标识值的分片就能最终正确的重装成为原来的数据报
- **标志位**字段占**3位**

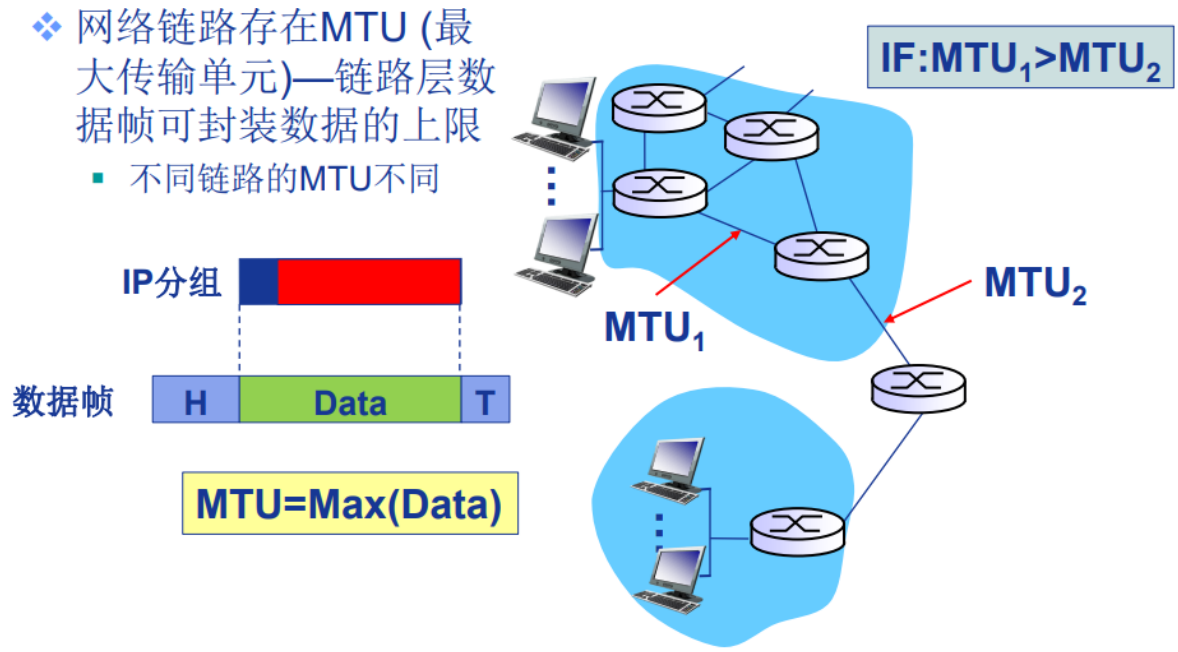


  - 目前只有两位有意义
  - 最低位记为MF
    - MF=1即表示后面"还有分片"的数据包
    - MF=0表示这已是若干数据包片中的最后一个
  - 中间位记为DF，意思是"不能分片"。只有当DF=0时才允许分片
- **片偏移**字段占**13位**：表示每个数据报的分片在原数据报中的相对位置
  - 片偏移以**8个字节**为偏移单位，即每个分片的长度一定是8字节的整数倍
- **生存时间（TTL）**字段占**8位**：IP分组在网络中可以通过的路由器数（或跳步数）

    - 路由器转发一次分组，TTL减1
    - 如果TTL=0，路由器则丢弃该IP分组
- **协议**字段占**8位**：指示IP分组封装的是哪个协议的数据包
    - 实现复用/分解
    - E.g：6为TCP，表示封装的为TCP段；17为UDP，表示封装的是 UDP数据报
- **首部校验和**字段占**16位**：实现对IP分组首部的差错检测
    - 计算校验和时，该字段置全0
    - 采用反码算数运算求和，和的反码作为首部校验和字段
    - 逐跳计算、逐跳校验
- **源IP地址、目的IP地址**字段**各占32位**：分别标识发送分组的源主机/路由器(网络接口)和接收分组的目的主机/路由器（网络接口）的IP地址
- **选项字段**占长度**可变**，**范围在1~40B之间**：携带安全、源 选路径、时间戳和路由记录等内容
    - 实际上很少被使用
- **填充字段**占长度**可变**，**范围在0~3B之间**：目的是补齐整个 首部，符合32位对齐，即保证首部长度是4字节的倍数

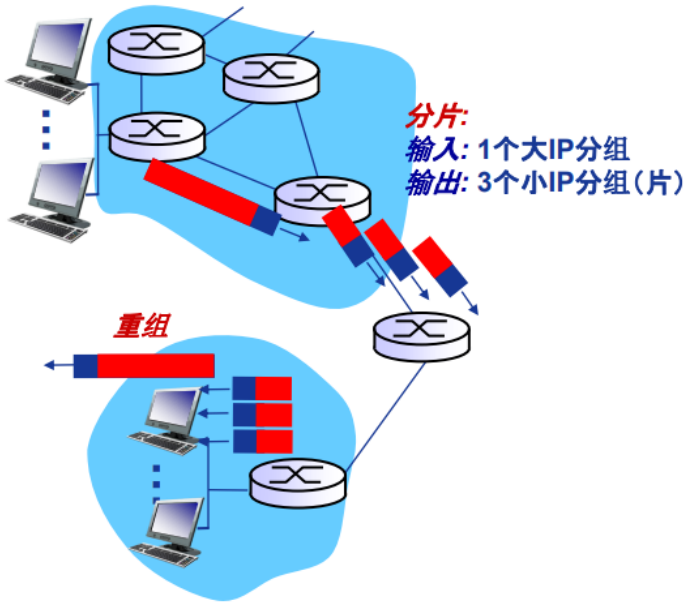
4.3.2 IP分片

最大传输单元 MTU



IP分片与重组

- 大IP分组向较小MTU链路转发时，可以被“分片” (fragmented)
  - 1个IP分组分为多片IP分组
  - IP分片到达目的的主机后进行“重组”(reassembled)
- IP首部的相关字段用于标识分片以及确定分片的相对顺序
  - 总长度、标识、标志位和片偏移



IP分片过程

- 假设原IP分组总长度为*L*，待转发链路的MTU为*M*
- 若*L*>*M*，且DF=0，则可以/需要分片
- 分片时每个分片的标识复制原IP分组的标识
- 通常分片时，除最后一个分片，其他分片均分为MTU允许的最大分片
- 一个最大分片可封装的数据应该是8的倍数，因此，一个最大分片可封装的数据为：

$$d = \left\lfloor \frac{M-20}{8} \right\rfloor \times 8$$

- 需要的总片数为：

$$n = \left\lceil \frac{L-20}{d} \right\rceil$$

❖ 每片的片偏移字段取值为：

$$F_i = \frac{d}{8} \times (i-1), \qquad 1 \le i \le n$$

❖ 每片的总长度字段为：

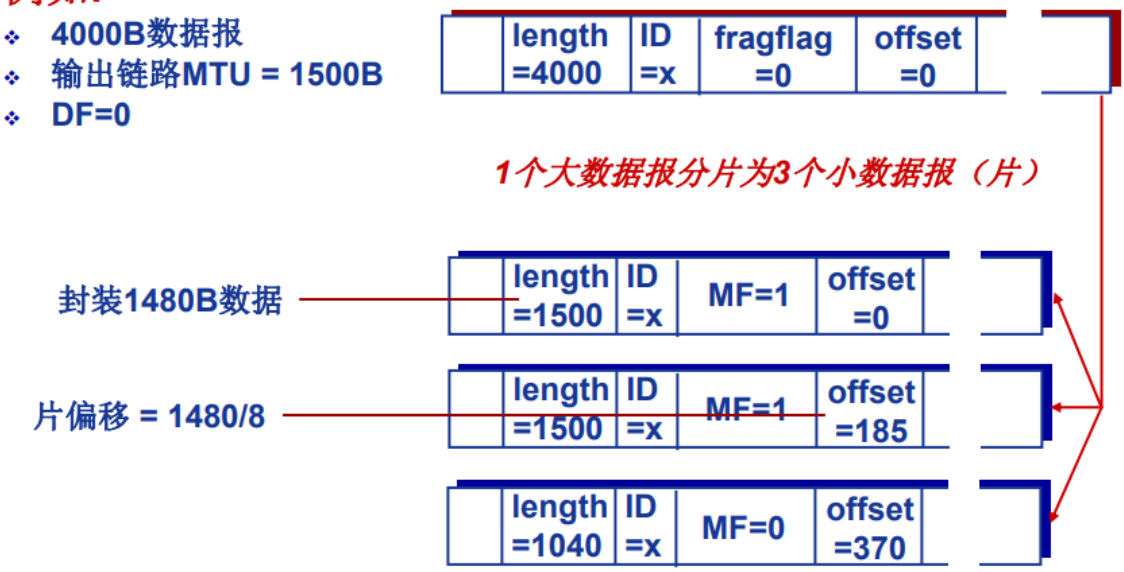
$$L_i = \begin{cases} d+20 & 1 \le i < n \\ L-(n-1)d & i = n \end{cases}$$

❖ 每片的MF标志位为：

$$\mathrm{MF}_i = \begin{cases} 1 & 1 \le i < n \\ 0 & i = n \end{cases}$$

例如：

- **4000B数据报**
- **输出链路MTU = 1500B**
- **DF=0**

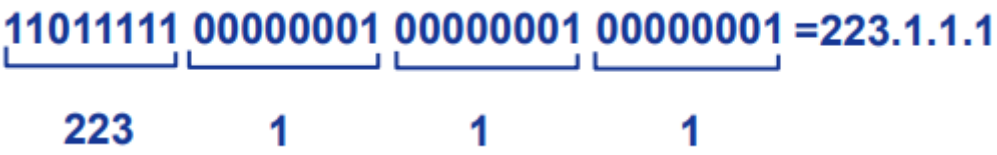


4.3.3 IP编址

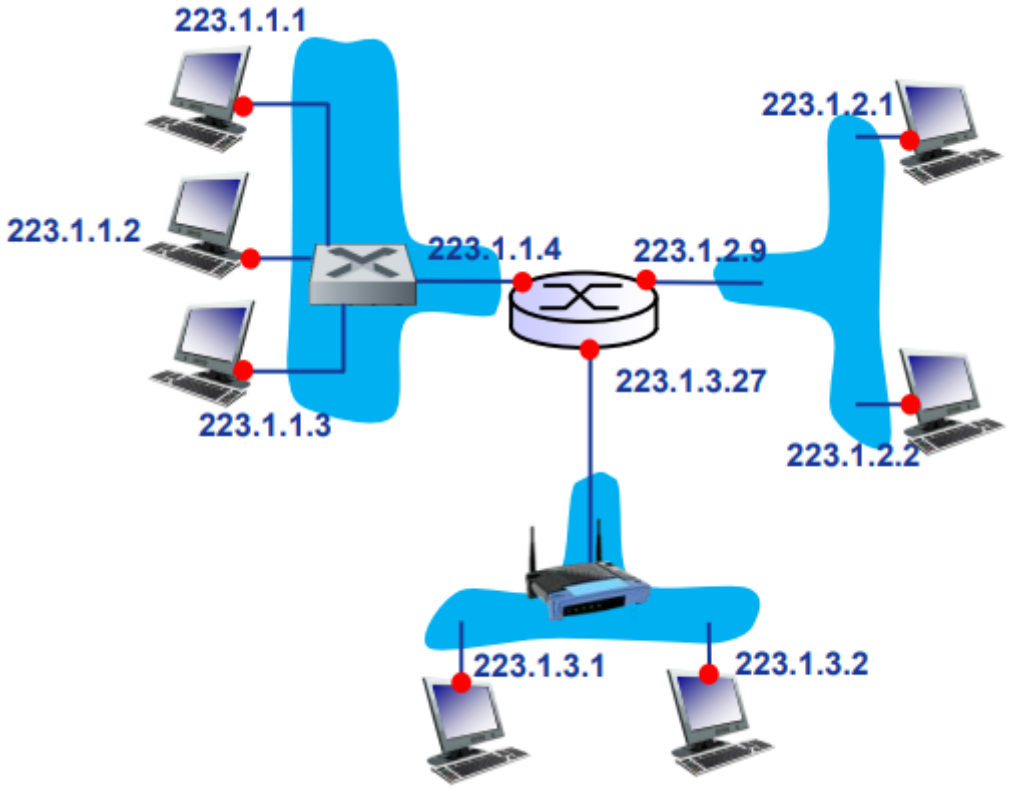
- IP分组
  - 源地址(SA)-从哪儿来
  - 目的地址(DA)-到哪儿去
- 接口(interface): 主机/路由器与物理链路的连接
  - 实现网络层功能
  - 实现网络层功能
  - 主机通常只有一个或两个 接口 (e.g：有线的以太网 接口，无线的802.11接口)

IP编址（addressing）

- IP地址: 32比特(IPv4) 编号标识主机、路由器的接口



- IP地址与每个接口关联



IP子网（Subnets）

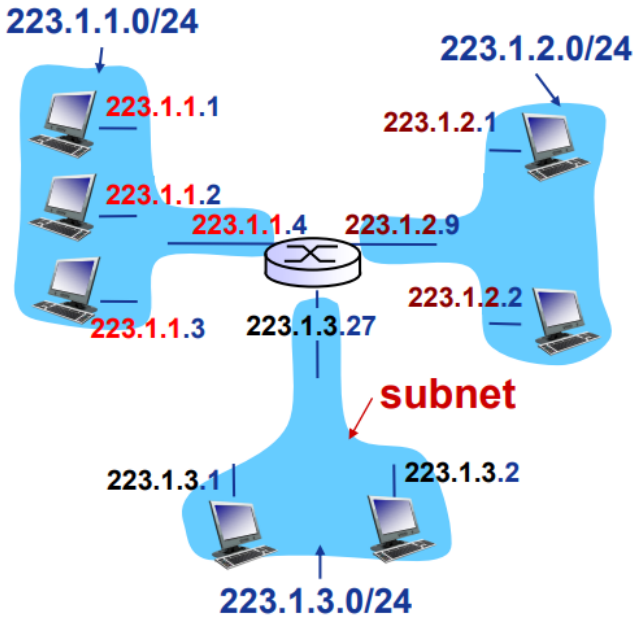
❖IP地址:

- 网络号(NetID) – 高位比特
- 主机号(HostID) – 低位比特

NetID	HostID

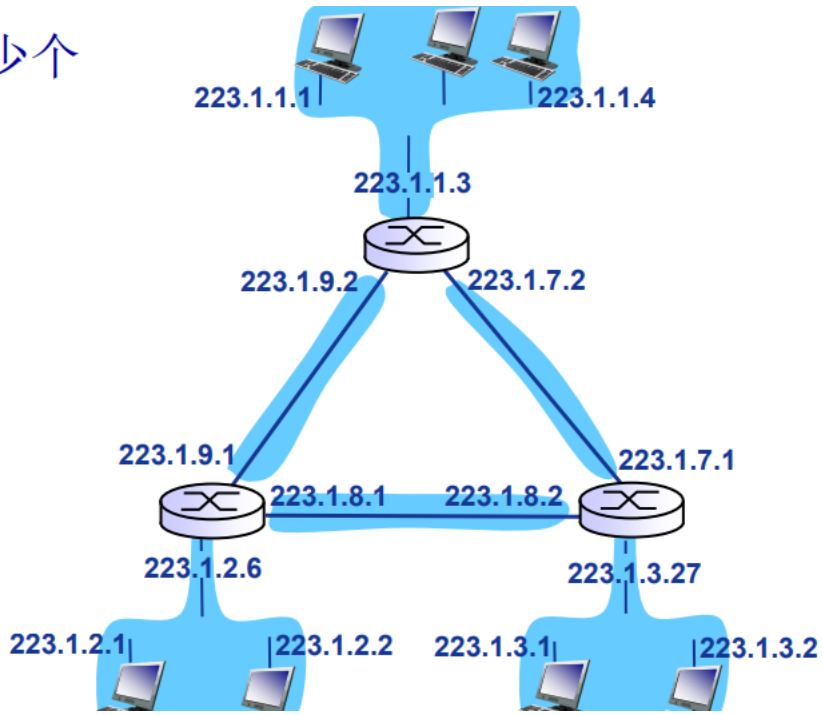
❖IP子网：

- IP地址具有相同网络号的设备接口
- 不跨越路由器（第三及以上层网络设备）可以彼此物理联通的接口



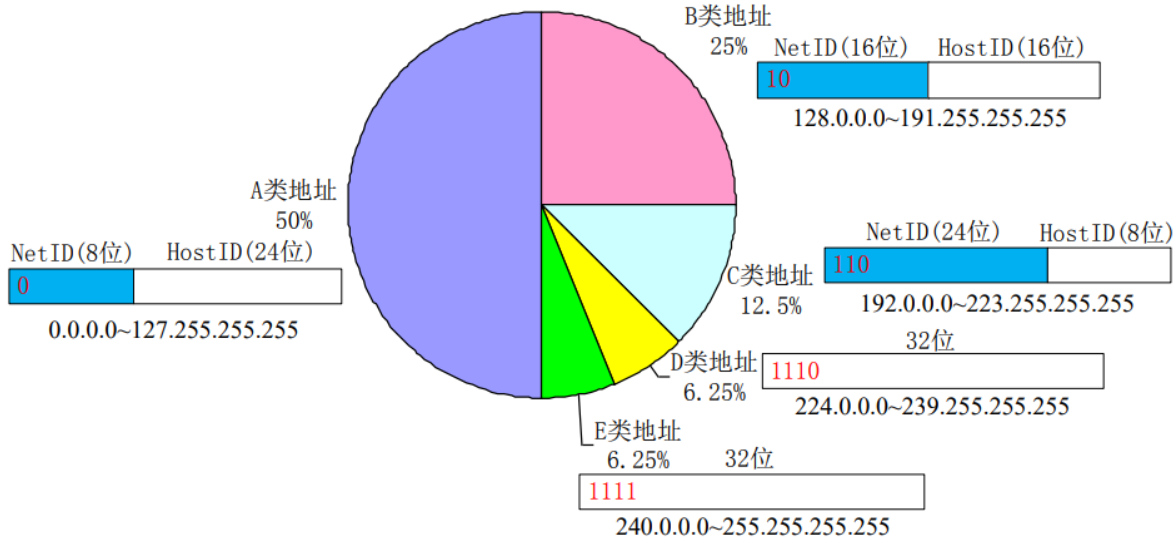
图中网络有多少个IP子网?

六个



4.3.4 有类IP地址

“有类”编址:



特殊IP地址

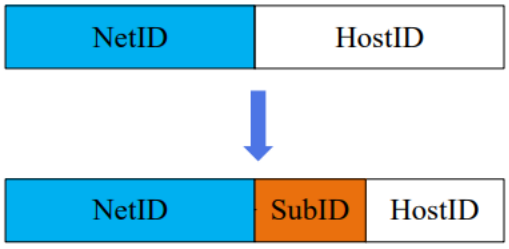
NetID	HostID	作为IP分组源地址	作为IP分组目的地址	用途
全0	全0	可以	不可以	在本网范围内表示本机；在路由表中用于表示默认路由(相当于表示整个Internet网络)
全0	特定值	不可以	可以	表示本网内某个特定主机
全1	全1	不可以	可以	本网广播地址(路由器不转发)
特定值	全0	不可以	不可以	网络地址，表示一个网络
特定值	全1	不可以	可以	直接广播地址，对特定网络上的所有主机进行广播
127	非全0或非全1的任何数	可以	可以	用于本地软件环回测试，称为环回地址

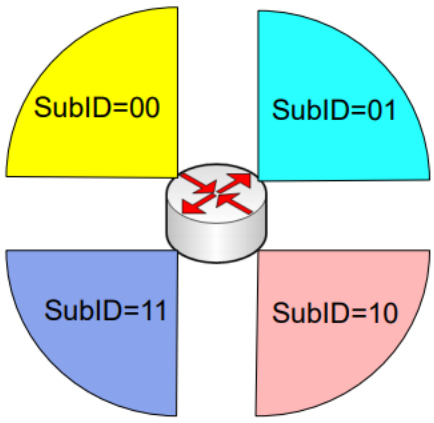
私有IP地址

Class	NetIDs	Blocks
A	10	1
B	172.16 to 172.31	16
C	192.168.0 to 192.168.255	256

4.3.5 IP子网划分与子网掩码

- IP地址:
  - 网络号(NetID) – 高位比特
  - 子网号(SubID) – 原网络主机号部分比特
  - 主机号(HostID) – 低位比特





- 如何确定是否划分了子网？利用多少位划分子网？
  - 子网掩码

子网掩码

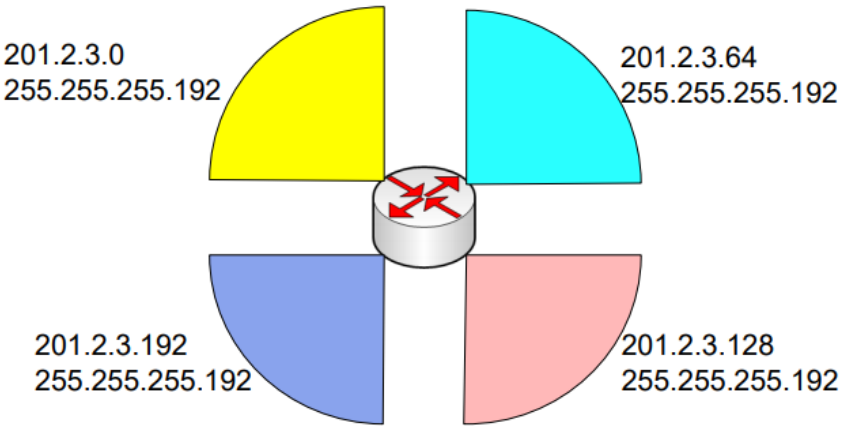
- 形如IP地址:
  - 32位
  - 点分十进制形式

- 取值：
  - NetID、SubID位全取1
  - HostID位全取0

子网地址+子网掩码
→准确确定子网大小

- 例如：
  - A网的默认子网掩码为：255.0.0.0
  - B网的默认子网掩码为：255.255.0.0
  - C网的默认子网掩码为：255.255.255.0
  - 借用3比特划分子网的B网的子网掩码为：255.255.224.0

- 例如：
  - 子网201.2.3.0，255.255.255.0，划分为等长的4个子网



- 路由器如何确定应该将IP分组转发到哪个子网？

❖将IP分组的目的IP地址与子网掩码**按位与**运算，提取子网地址

❖例如：

- 目的IP地址：172.32.1.112，子网掩码：255.255.254.0

172. 32. 1. 112=	10101100	00100000	00000001	01110000
255. 255. 254. 0=	11111111	11111111	11111110	00000000
	10101100	00100000	00000000	00000000
	172	32	0	0

- 子网地址：172.32.0.0(子网掩码：255.255.254.0)
- 地址范围：172.32.0.0~172.32.1.255
- 可分配地址范围：172.32.0.1~172.32.1.254
- 广播地址：172.32.1.255

C类网络子网划分举例

（红、绿都不能作为主机IP，红色表示本网络，绿色表示本网络广播号）

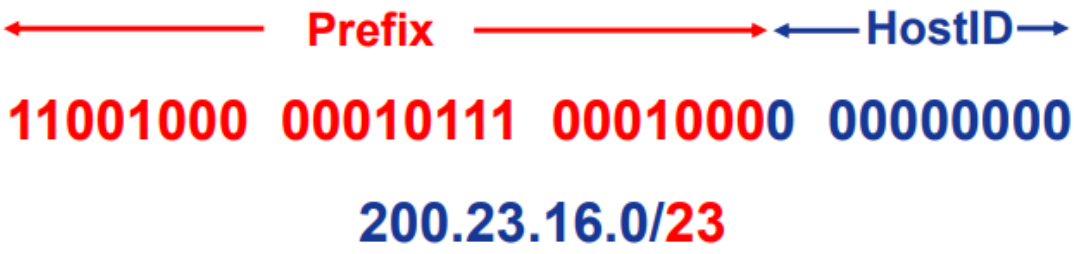
子网	SubID (二进制)	HostID取值范围 (二进制)	第4八位组取值范围 (十进制)
1#	000	00000 thru 11111	.0 thru .31
2#	001	00000 thru 11111	.32 thru .63
3#	010	00000 thru 11111	.64 thru .95
4#	011	00000 thru 11111	.96 thru .127
5#	100	00000 thru 11111	.128 thru .159
6#	101	00000 thru 11111	.160 thru .191
7#	110	00000 thru 11111	.192 thru .223
8#	111	00000 thru 11111	.224 thru .255

4.4 CIDR与路由聚集

CIDR

无类域间路由(CIDR: Classless InterDomain Routing)

- 消除传统的A类、B类和C类地址界限
  - NetID+SubID→Network Prefix (Prefix)可以任意长度
- 融合子网地址与子网掩码，方便子网划分
  - 无类地址格式：a.b.c.d/x，其中x为前缀长度
- 例如

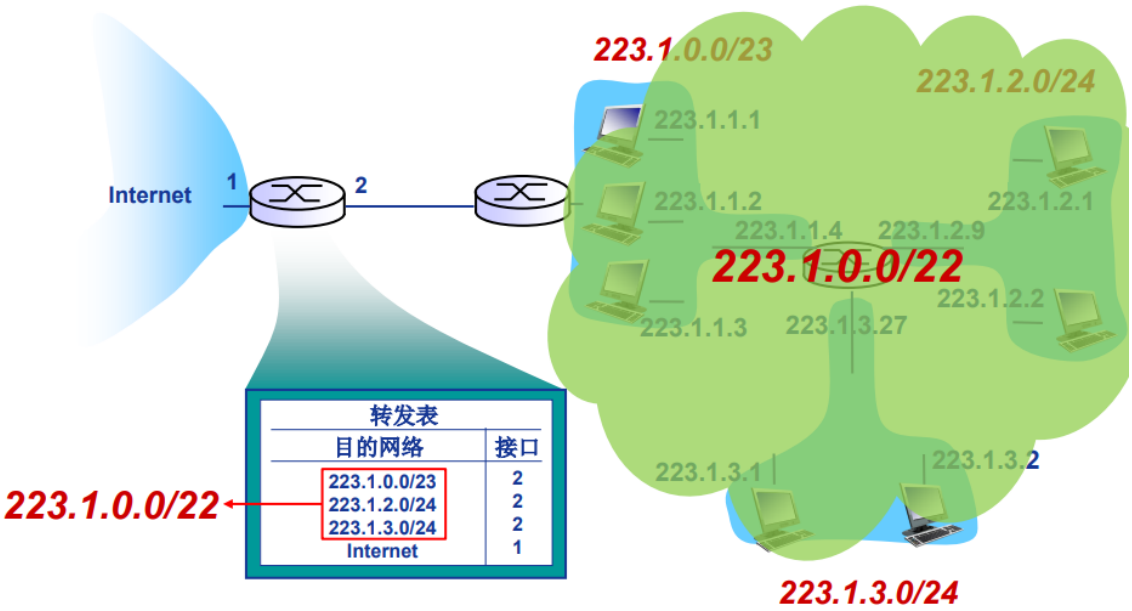


- 子网201.2.3.64，255.255.255.192→201.2.3.64/26

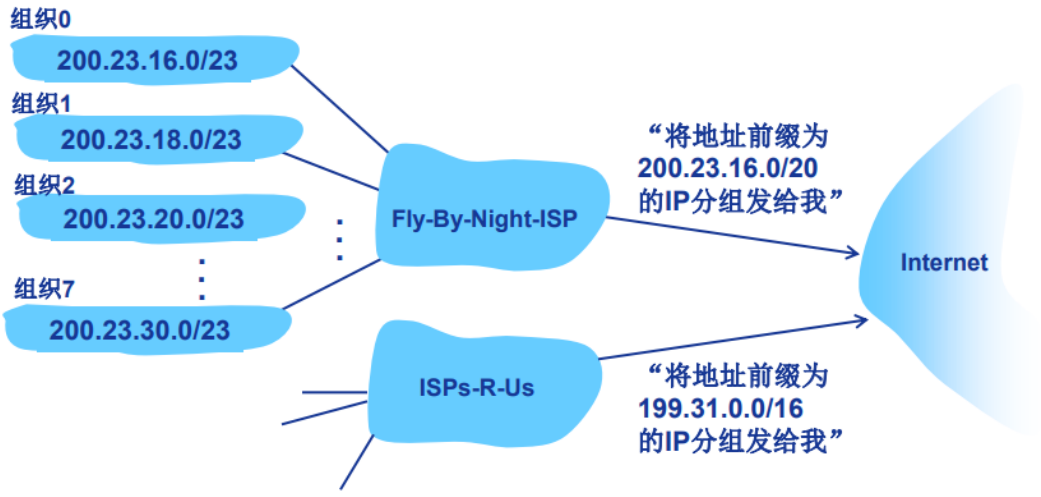
CIDR与路由聚合

- 提高IPv4 地址空间分配效率
- 提高路由效率
  - 将多个子网聚合为一个较大的子网
  - 构造超网（supernetting）
  - 路由聚合（route aggregation）

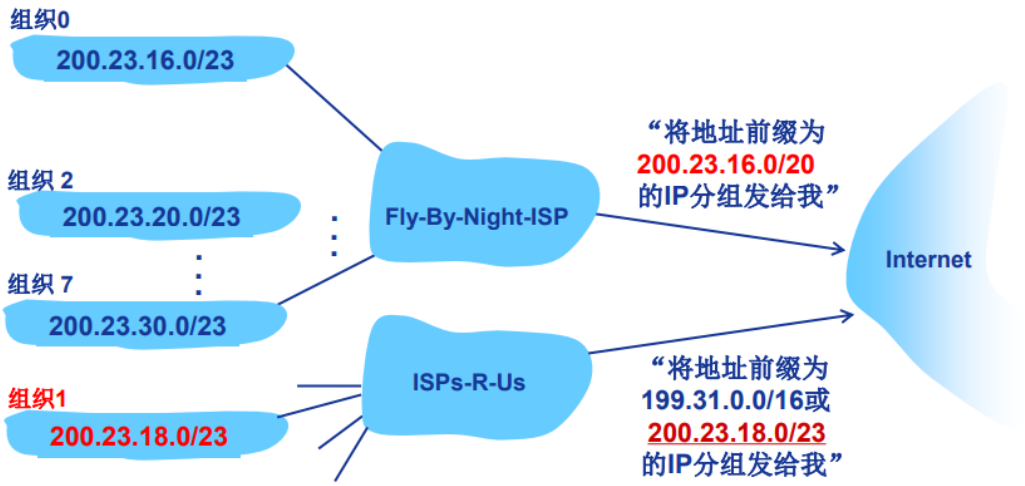
路由聚合

- 

层级编址使得路由信息通告更高效：

- 

选用更具体的路由：最长前缀匹配优先！

- 

4.5 DHCP协议

如何获得IP地址?

- 硬编码

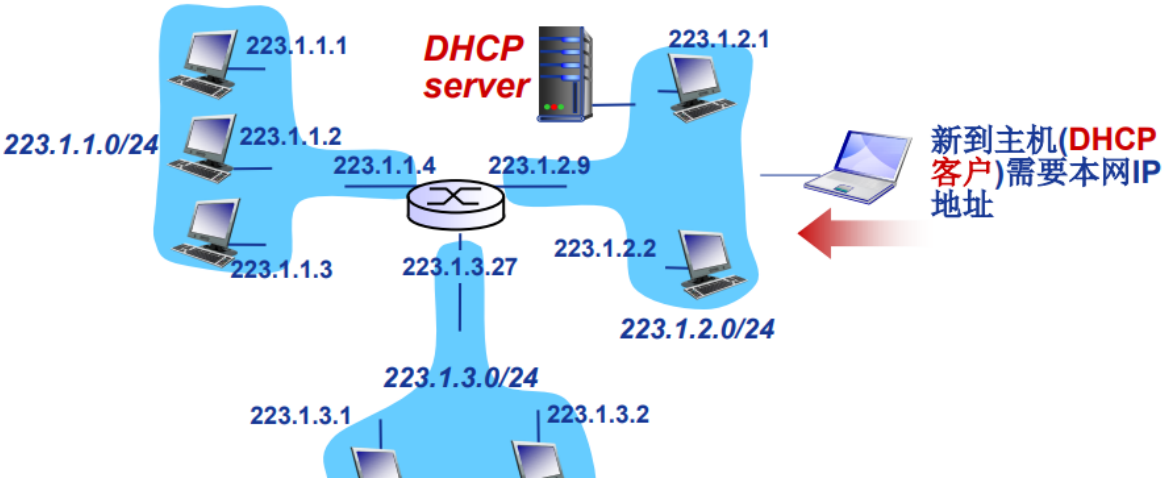
❖ “硬编码”

▪ 静态配置

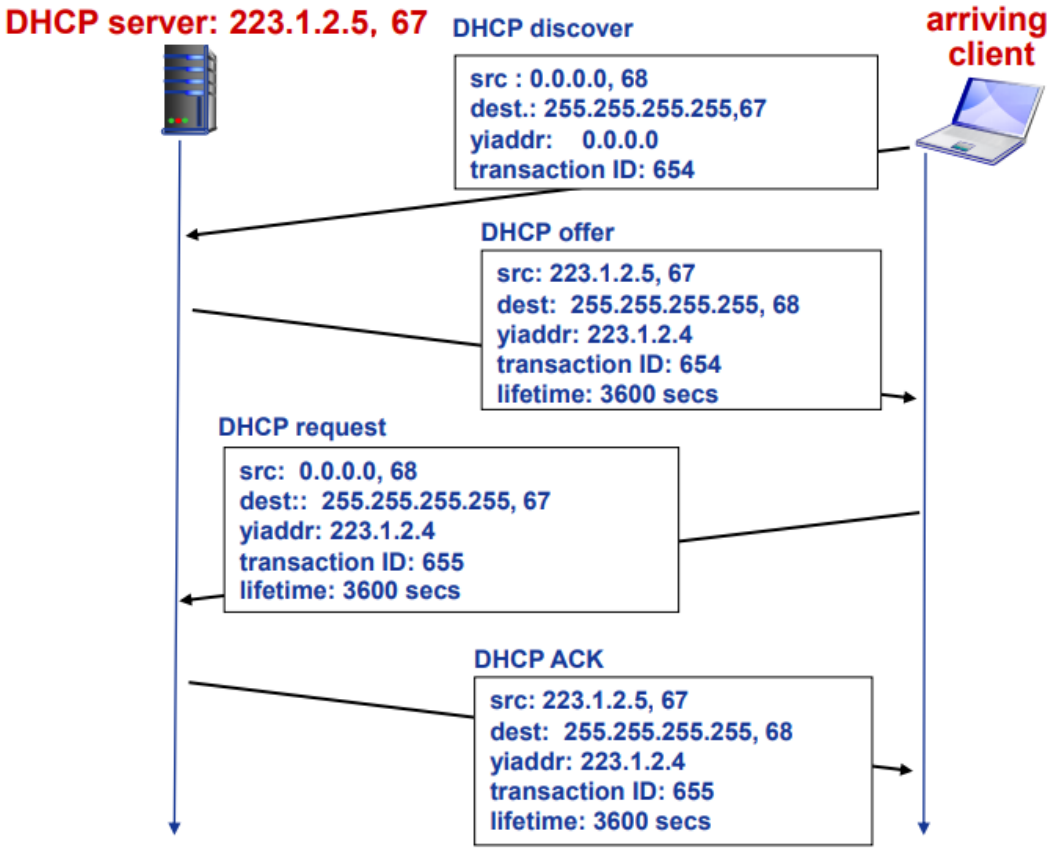
- 动态主机配置协议-DHCP: Dynamic Host Configuration Protocol
  - 从服务器动态获取：
    - IP地址
    - 子网掩码
    - 默认网关地址
    - DNS服务器名称与IP地址
  - 即插即用
  - 允许地址重用
  - 支持在用地址续租
  - 支持移动用户加入网络

动态主机配置协议(DHCP)

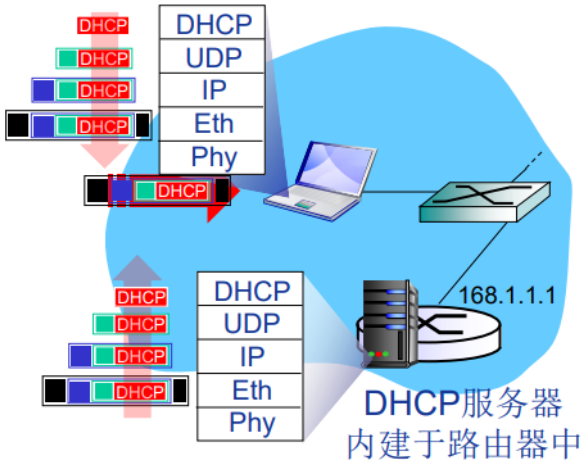
- 主机广播 “DHCP discover”(发现报文)
- DHCP服务器利用 “DHCP offer” (提供报文) 进行响应
- 主机请求IP地址: “DHCP request” (请求报文)
- DHCP服务器分配IP地址: “DHCP ack” (确认报文)



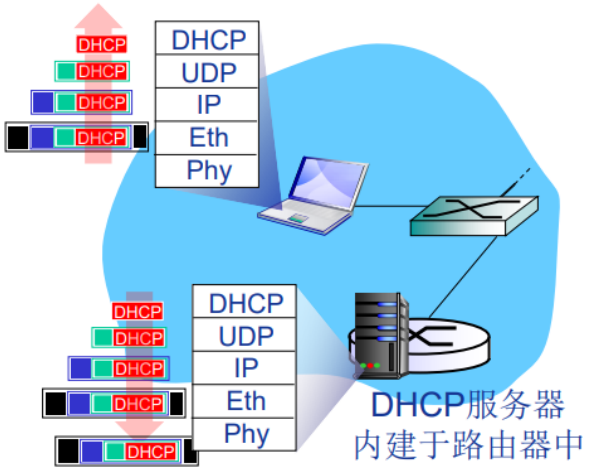
举例



- ❖DHCP协议在应用层实现
  - 请求报文封装到UDP数据报中
  - IP广播
  - 链路层广播 (e.g. 以太网广播)

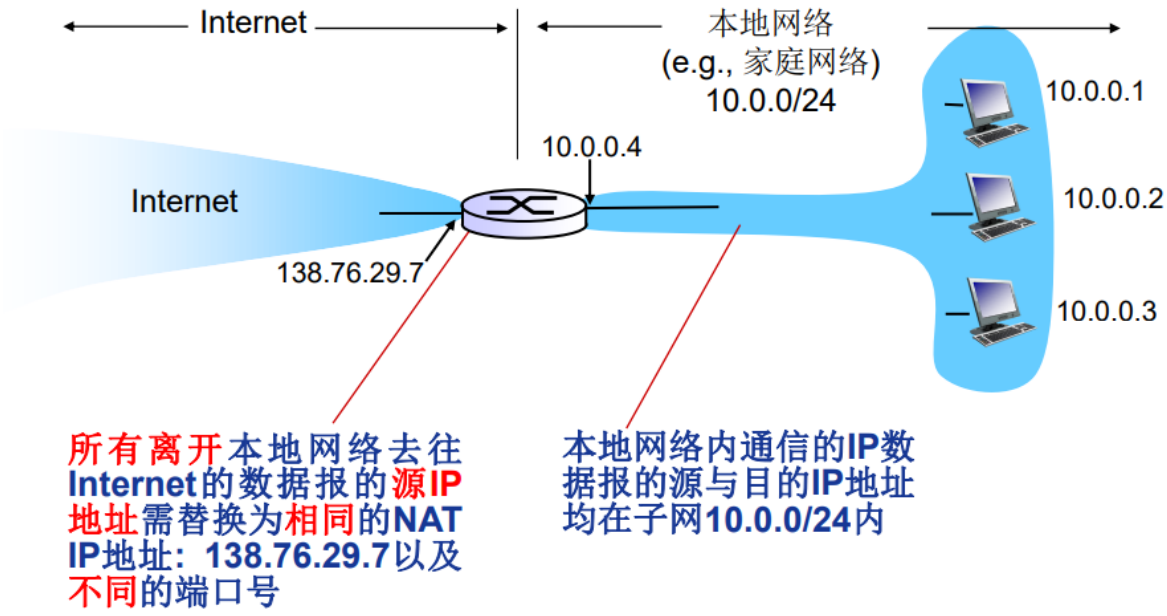


- ❖DHCP服务器构造ACK报文
  - 包括分配给客户的IP地址、子网掩码、默认网关、DNS服务器地址



4.6 NAT

网络地址转换NAT



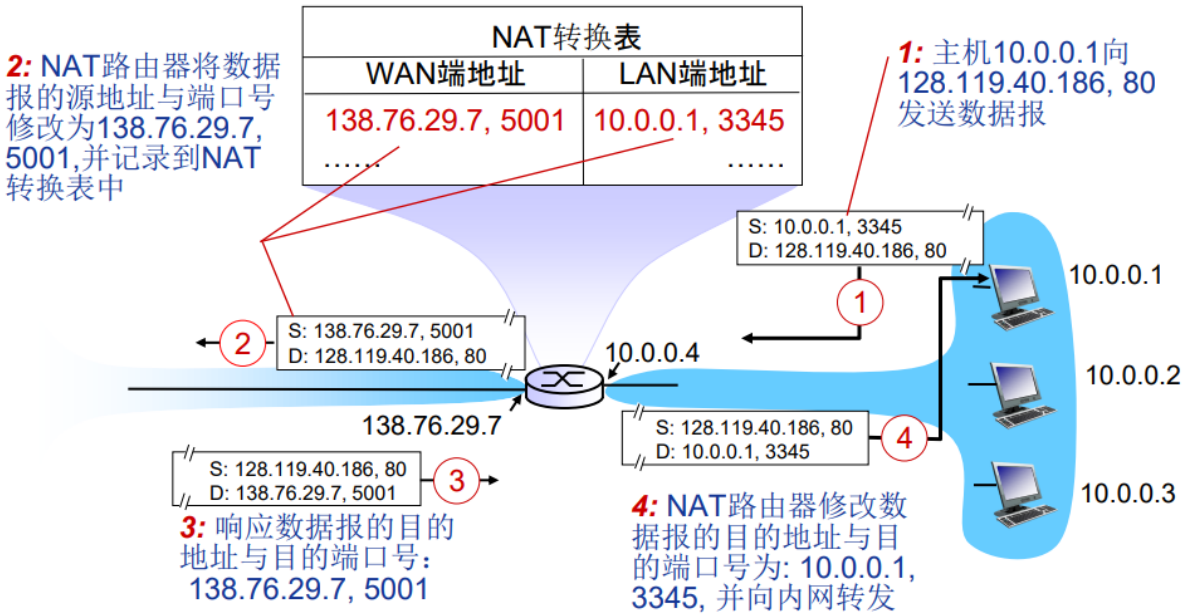
动机

- 只需/能从ISP申请一个IP地址
  - IPv4地址耗尽
- 本地网络设备IP地址的变更，无需通告外界网络
- 变更ISP时，无需修改内部网络设备IP地址
- 内部网络设备对外界网络不可见，即不可直接寻址(安全)

实现（替换→记录→替换）

- 替换
  - 利用(NAT IP地址,新端口号)替换每个外出IP数据报的(源IP地址,源端口号
- 记录
  - 将每对(NAT IP地址, 新端口号) 与(源IP地址, 源端 口号)的替换信息存储到NAT转换表中
- 替换
  - 根据NAT转换表，利用(源IP地址, 源端口号)替换每个进入内网IP数据报的(目的IP地址,目的端口号)， 即(NAT IP地址, 新端口号)

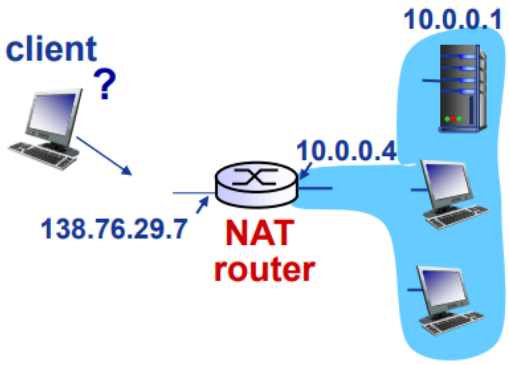
NAT过程



- 16-bit端口号字段:
  - 可以同时支持60,000多并行连接！
- NAT主要争议:
  - 路由器应该只处理第3层功能
  - 违背端到端通信原则
    - 应用开发者必须考虑到NAT的存在，e.g., P2P应用
  - 地址短缺问题应该由IPv6来解决

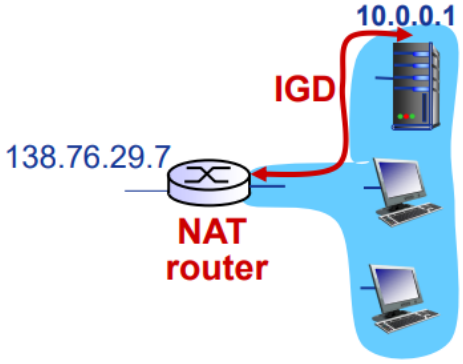
NAT穿透问题

- 客户期望连接内网地址为10.0.0.1的服务器
  - 客户不能直接利用地址10.0.0.1直接访问服务器
  - 对外唯一可见的地址是NAT地址: 138.76.29.7

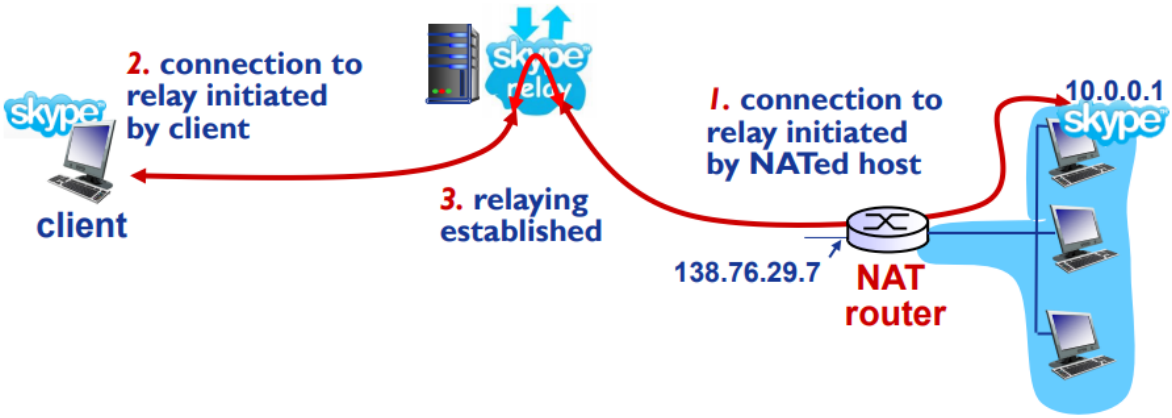


- 解决方案1: 静态配置NAT，将特定端口的连接请求转发给服务器
  - e.g., (138.76.29.7, 2500) 总是转发给(10.0.0.1, 25000)

- 解决方案2: 利用UPnP (Universal Plug and Play) 互联网网关设备协议 (IGD-Internet Gateway Device) 自动配置:
  - 学习到NAT公共IP地址(138.76.29.7)
  - 在NAT转换表中，增删端口映射



- 解决方案3: 中继(如Skype)
  - NAT内部的客户与中继服务器建立连接
  - 外部客户也与中继服务器建立连接
  - 中继服务器桥接两个连接的分组



4.7 ICMP协议

互联网控制报文协议 ICMP (Internet Control Message Protocol)支持主机或路由器：

- 差错(或异常)报告
- 网络探询

两类ICMP 报文:

- 差错报告报文(5种)
    - 目的不可达 - 无法成功交付给所要交付的目的，丢弃，向源主机发送
    - 源抑制(Source Quench) - 缓存满，导致到达的数据丢弃，向源主机发送，令其降低发送速度
    - 超时/超期 - TTL减为0，向源主机发送
    - 参数问题 - 数据报某些字段发生差错，向源主机发送
    - 重定向 (Redirect) - 不应该经过本路由，向源主机发送
- 网络探询报文(2组)
    - 回声(Echo)请求与应答报文(Reply) - 探测网络是否可达；PING
    - 时间戳请求与应答报文

类型(Type)	编码(Code)	description
0	0	回声应答 (ping)
3	0	目的网络不可达
3	1	目的主机不可达
3	2	目的协议不可达
3	3	目的端口不可达
3	6	目的网络未知
3	7	目的主机未知
4	0	源抑制(拥塞控制-未用)
8	0	回声请求(ping)
9	0	路由通告
10	0	路由发现
11	0	TTL超期
12	0	IP首部错误

例外情况

几种不发送ICMP差错报告报文的特殊情况：

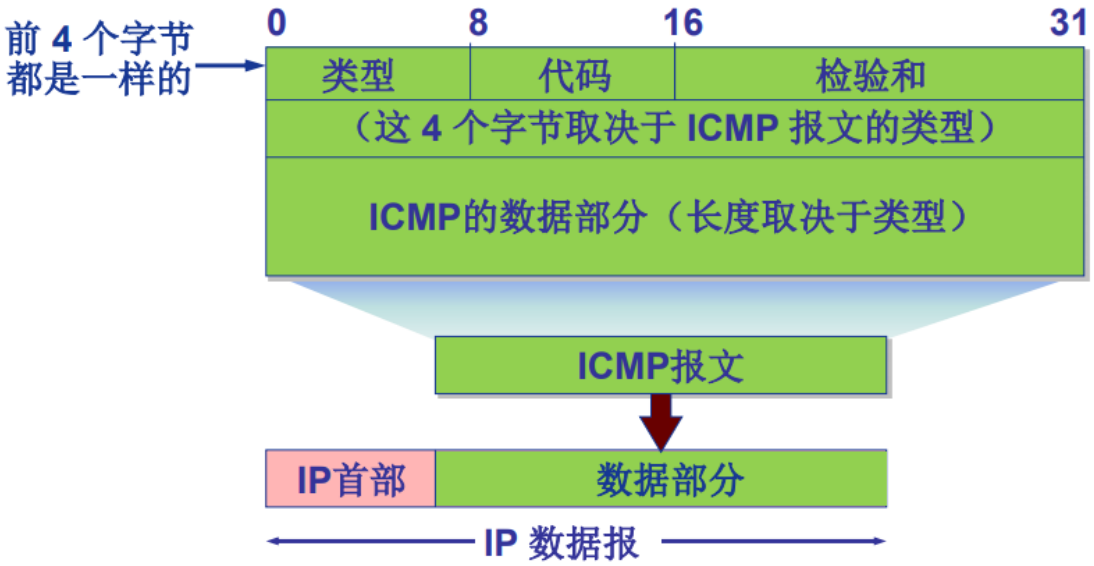
- 对ICMP差错报告报文不再发送ICMP差错报告报文
- 除第1个IP数据报分片外，对所有后续分片均不发送ICMP差错报告报文
- 对所有多播IP数据报均不发送 ICMP差错报告报文
- 对具有特殊地址（如127.0.0.0 或 0.0.0.0）的IP数据报不发送ICMP差错报告报文

几种不再使用的ICMP报文

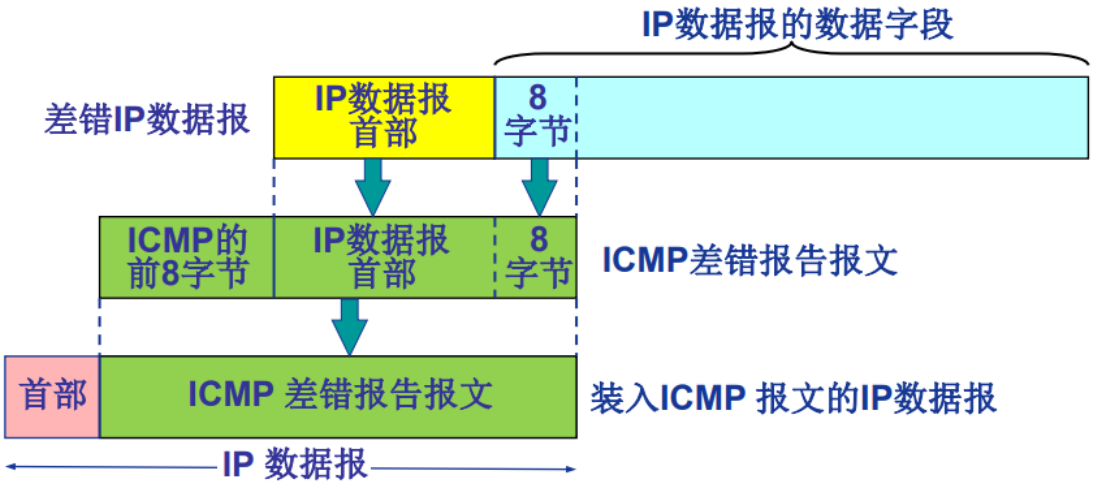
- 信息请求与应答报文
- 子网掩码请求和应答报文
- 路由器询问和通告报文

ICMP报文格式

- ICMP报文封装到IP数据报中传输



ICMP差错报告报文的封装



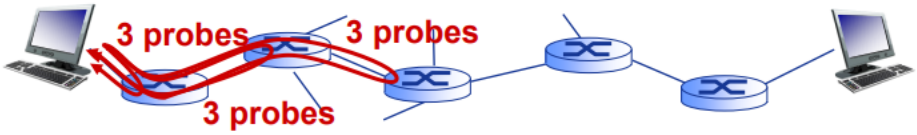
ICMP应用举例：Traceroute

- 源主机向目的主机发送一系列UDP数据报
  - 第1组IP数据报TTL =1
  - 第2组IP数据报TTL=2, etc.
  - 目的端口号为不可能使用的端口号
- 当第*n*组数据报(TTL=*n*)到达第*n*个路由器时：
  - 路由器丢弃数据报
  - 向源主机发送ICMP报文(type=11, code=0)
  - ICMP报文携带路由器名称和IP地址信息

- 当ICMP报文返回到源主机时，记录RTT

停止准则:

- **UDP数据报最终到达目的主机**
- **目的主机返回“目的端口不可达”ICMP报文 (type=3, code=3)**
- **源主机停止**



4.8 IPv6

出现IPV6的动机

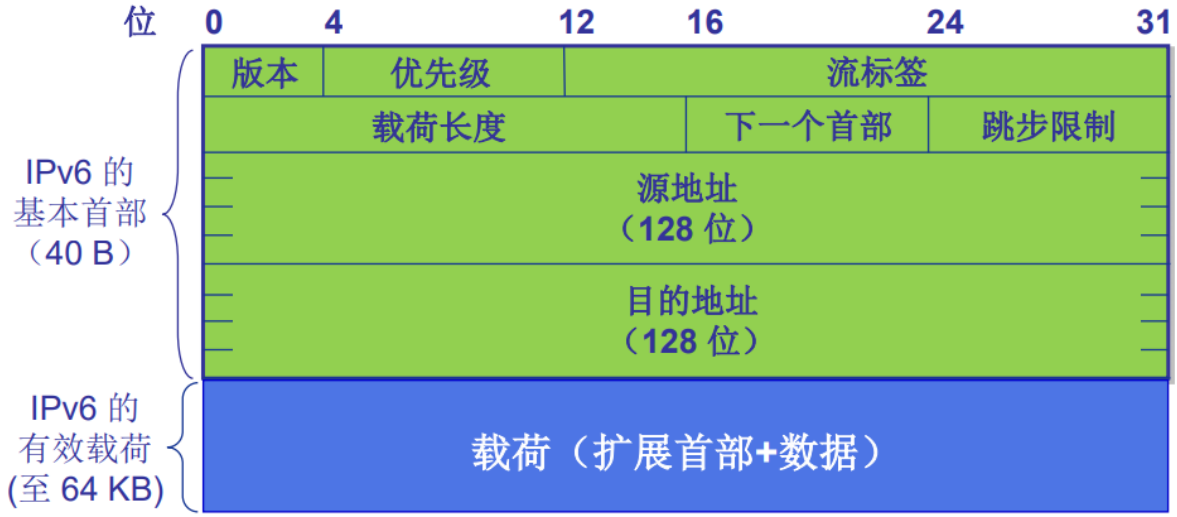
- 最初动机: 32位IPv4地址空间已分配殆尽
- 其他动机: 改进首部格式
  - 快速处理/转发数据报
  - 支持QoS

IPv6数据报格式:

- 固定长度的40字节基本首部
- 不允许分片

IPv6数据报格式

优先级(priority): 标识数据报的优先级
流标签(flow Label): 标识同一“流”中的数据报
下一个首部(next header): 标识下一个选项首部或上层协议首部(如TCP首部)



相对于IPv4的其它改变

- 校验和(checksum): 彻底移除，以减少每跳处理时间
- 选项(options): 允许，但是从基本首部移出，定义多个选项首部，通过“下一个首部”字段指示
- ICMPv6: 新版ICMP
  - 附加报文类型，e.g：“Packet Too Big”
  - 多播组管理功能

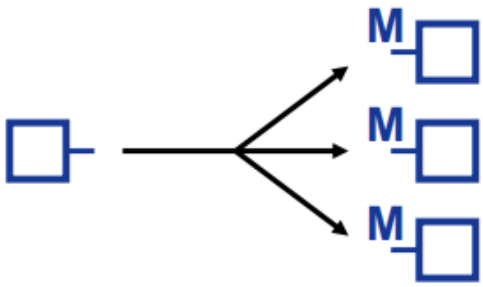
IPv6地址表示

- 一般形式: **1080:0:FF:0:8:800:200C:417A**
- 压缩形式：**FF01:0:0:0:0:0:0:43 → FF01::43**
- IPv4嵌入形式：**0:0:0:0:0:FFFF:13.1.68.3**或**::FFFF:13.1.68.3**
- 地址前缀：**2002:43c:476b::/48**
  - IPv6**不再使用**掩码
- URLs：**http://[3ffe::1:800:200c:417a]:8000**

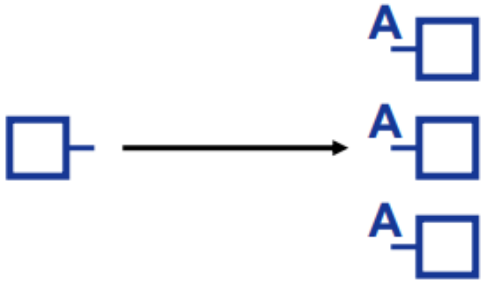
IPv6基本地址类型

单播(unicast):
一对一通信

多播(multicast):
一对多通信

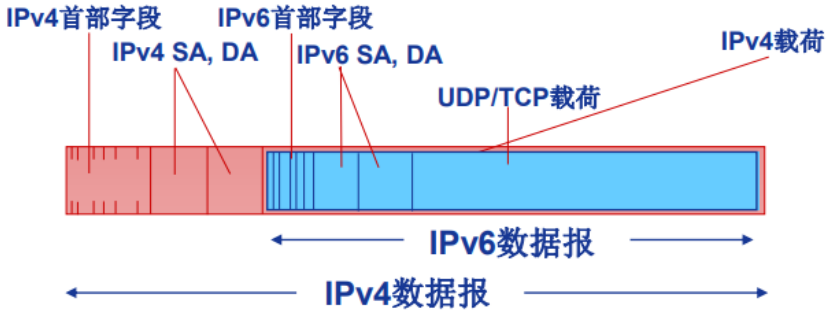


任意播(anycast):
一对一组之一
（最近一个）通信

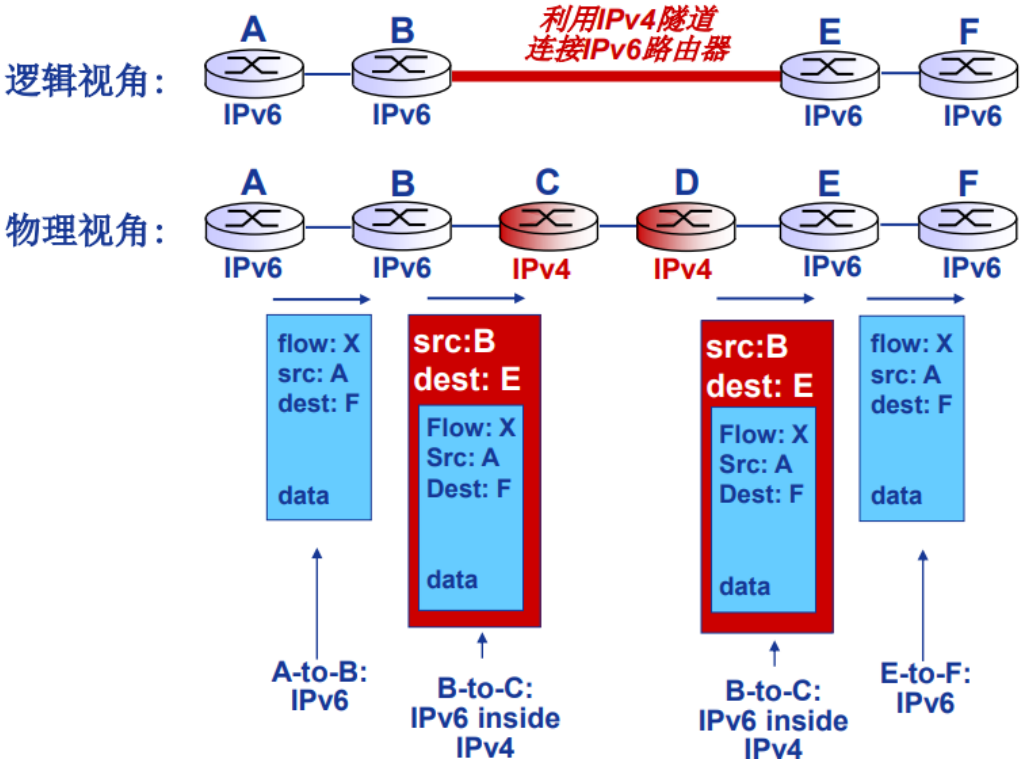


IPv4向IPv6过渡

- 不可能在某个时刻所有路由器同时被更新为IPv6
  - 不会有 “标志性的日期”
  - IPv4和IPv6路由器共存的网络如何运行？
- 隧道(tunneling): IPv6数据报作为IPv4数据报的载荷进行封装，穿越IPv4网络



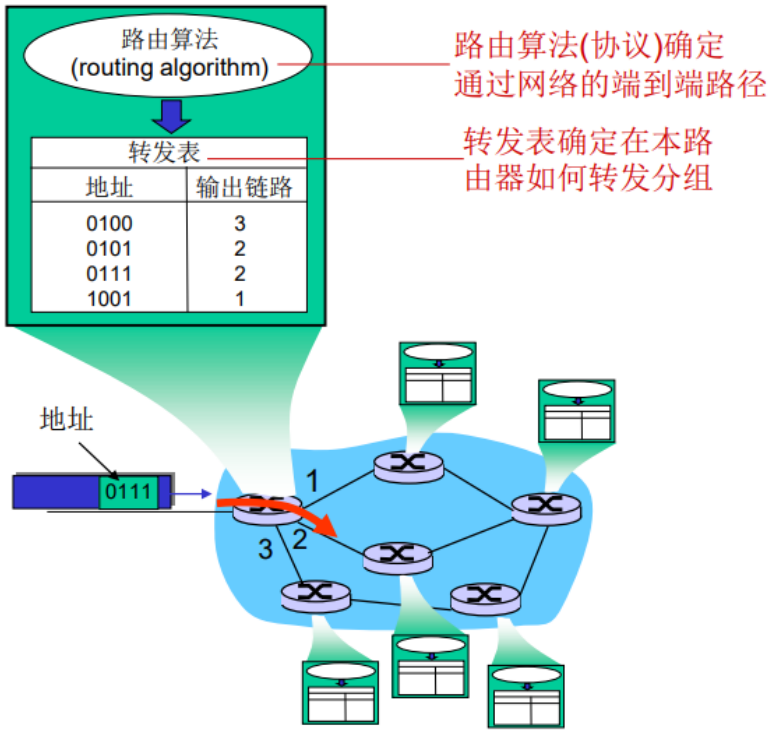
隧道



4.9 路由算法

路由与转发

- 转发(forwarding): 将分组从路由器的输入端口转移到合适的输出端口
- 路由(routing): 确定分组从源到目的经过的路径
  - 路由算法（routing algorithms）



网络抽象：图

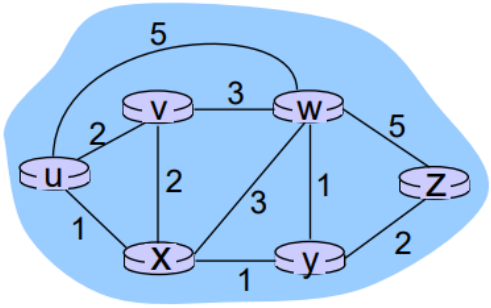


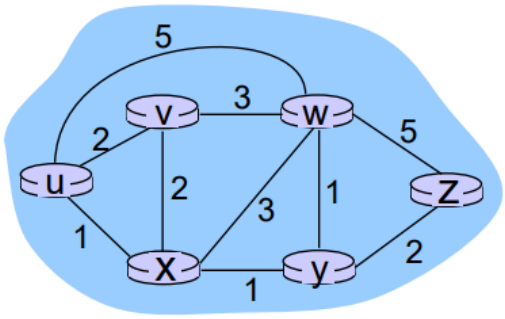
图: G = (N, E)

N = 路由器集合= { u, v, w, x, y, z }

E = 链路集合 ={ (u,v), (u,x), (v,x), (v,w), (x,w), (x,y), (w,y), (w,z), (y,z) }

附注: 图的抽象在网络领域应用很广泛
E.g.:P2P，其中，N是 peers集合，而E是TCP连接集合

图抽象：费用



c(x, x') = 链路(x, x')的费用
e.g., c(w, z) = 5

每段链路的费用可以总是1，

或者是，

带宽的倒数、拥塞程度等

路径费用：$(x_1, x_2, x_3, \ldots, x_p) = c(x_1,x_2) + c(x_2,x_3) + \ldots + c(x_{p-1},x_p)$

关键问题: 源到目的（如u到z）的最小费用路径是什么？
路由算法: 寻找最小费用路径的算法

路由算法分类

静态路由 vs 动态路由？

静态路由：

- 手工配置
- 路由更新慢
- 优先级高

动态路由：

- 路由更新快
  - 定期更新
  - 及时响应链路费用或网络拓扑变化

全局信息 vs 分散信息？

全局信息：

- 所有路由器掌握完整的网络拓扑和链路费用信息
- **E.g. 链路状态(LS)路由算法**

分散(decentralized)信息：

- 路由器只掌握物理相连的邻居以及链路费用
- 邻居间信息交换、运算的迭代过程
- **E.g. 距离向量(DV)路由算法**

4.9.1 链路状态路由算法

Dijkstra 算法

- 所有结点(路由器)掌握网络拓扑和链路费用
  - 通过“链路状态广播”
  - 所有结点拥有相同信息
- 计算从一个结点(“源”)到达所有其他结点的最短路径
  - 获得该结点的转发表
- 迭代: k次迭代后，得到到达k个目的结点的最短路径

符号:

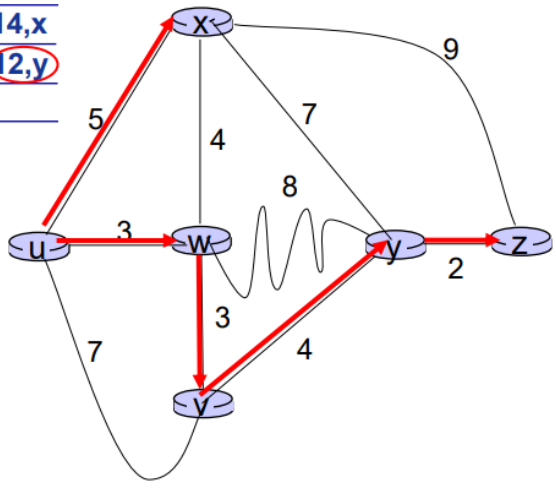
- c(x,y): 结点x到结点y链路费用；如果x和y不直接相连，则=∞
- D(v): 从源到目的v的当前路径费用值
- p(v): 沿从源到v的当前路径，v的前序结点
- N’: 已经找到最小费用路径的结点集合

```
1  初始化:
2   N' = {u}
3   for 所有结点v
4    if v毗邻u
5       then D(v) = c(u,v)
6    else D(v) = ∞
7
8  Loop
9    找出不在 N'中的w ， 满足D(w)最小
10   将w加入N'
11   更新w的所有不在N'中的邻居v的D(v) :
12     D(v) = min( D(v), D(w) + c(w,v) )
13   /*到达v的新费用或者是原先到达v的费用，或者是
14     已知的到达w的最短路径费用加上w到v的费用 */
15 until 所有结点在N'中
```

dijkstra算法例

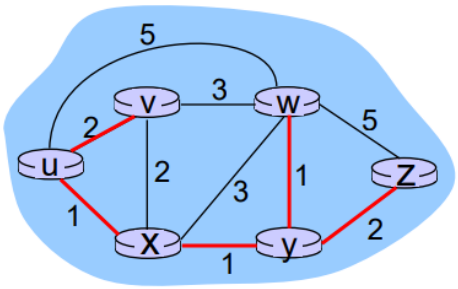
1.

Step	N'	D(v) p(v)	D(w) p(w)	D(x) p(x)	D(y) p(y)	D(z) p(z)
0	u	7,u	3,u	5,u	∞	∞
1	uw	6,w		5,u	11,w	∞
2	uwx	6,w			11,w	14,x
3	uwxv				10,v	14,x
4	uwxvy					12,y
5	uwxvyz					

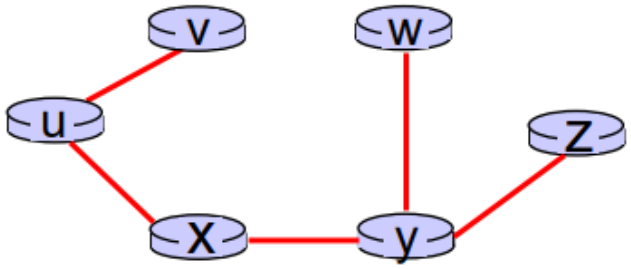


2.

Step	N'	D(v),p(v)	D(w),p(w)	D(x),p(x)	D(y),p(y)	D(z),p(z)
0	u	2,u	5,u	1,u	∞	∞
1	ux	2,u	4,x		2,x	∞
2	uxy	2,u	3,y			4,y
3	uxyv		3,y			4,y
4	uxyvw					4,y
5	uxyvwz					



u的最终最短路径树:



u的最终转发表:

目的	链路
v	(u,v)
x	(u,x)
y	(u,x)
w	(u,x)
z	(u,x)

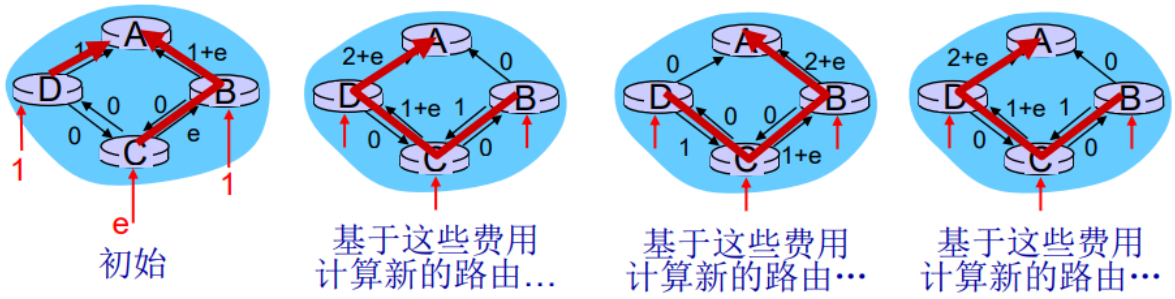
关于dijkstra算法的讨论

算法复杂性: n个结点

- 每次迭代: 需要检测所有不在集合N'中的结点w
- n(n+1)/2次比较: $O(n^2)$
- 更高效的实现: O(nlogn)

存在震荡(oscillations)可能:

- e.g., 假设链路费用是该链路承载的通信量:



4.9.2 距离向量路由算法

Bellman-Ford方程(动态规划)

令:

$d_x(y)$:=从x到y最短路径的费用（距离）

则:

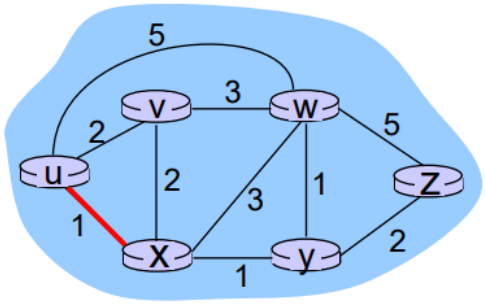
$$d_x(y) = \min_v \{c(x,v) + d_v(y)\}$$

从邻居v到达目的y的费用（距离）

x到邻居v的费用

在x的所有邻居v中取最小值

Bellman-Ford 举例



显然: $d_v(z) = 5$, $d_x(z) = 3$, $d_w(z) = 3$

根据B-F方程:

$$d_u(z) = \min \{ c(u,v) + d_v(z), \; c(u,x) + d_x(z), \; c(u,w) + d_w(z) \}$$
$$= \min \{2 + 5, \; 1 + 3, \; 5 + 3\} = 4$$

重点：结点获得最短路径的下一跳，该信息用于转发表中！

距离向量路由算法

- $D_x(y)$ = 从结点x到结点y的最小费用估计
  - x维护距离向量(DV): $\mathbf{D}_x = [D_x(y): y \in N]$
- 结点x:
  - 已知到达每个邻居的费用: c(x,v)
  - 维护其所有邻居的距离向量: $\mathbf{D}_v = [D_v(y): y \in N]$

核心思想:

- 每个结点不定时地将其自身的DV估计发送给其邻居
- 当x接收到邻居的新的DV估计时，即依据B-F更新其自身的距离向量估计：

$$D_x(y) \leftarrow \min_v \{c(x,v) + D_v(y)\} \quad \text{for each node } y \in N$$

- $D_x(y)$将最终收敛于实际的最小费用 $d_x(y)$

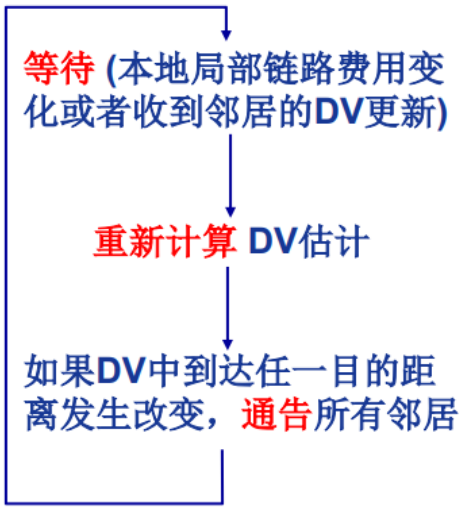
异步迭代:

- 引发每次局部迭代的因素
  - 局部链路费用改变
  - 来自邻居的DV更新

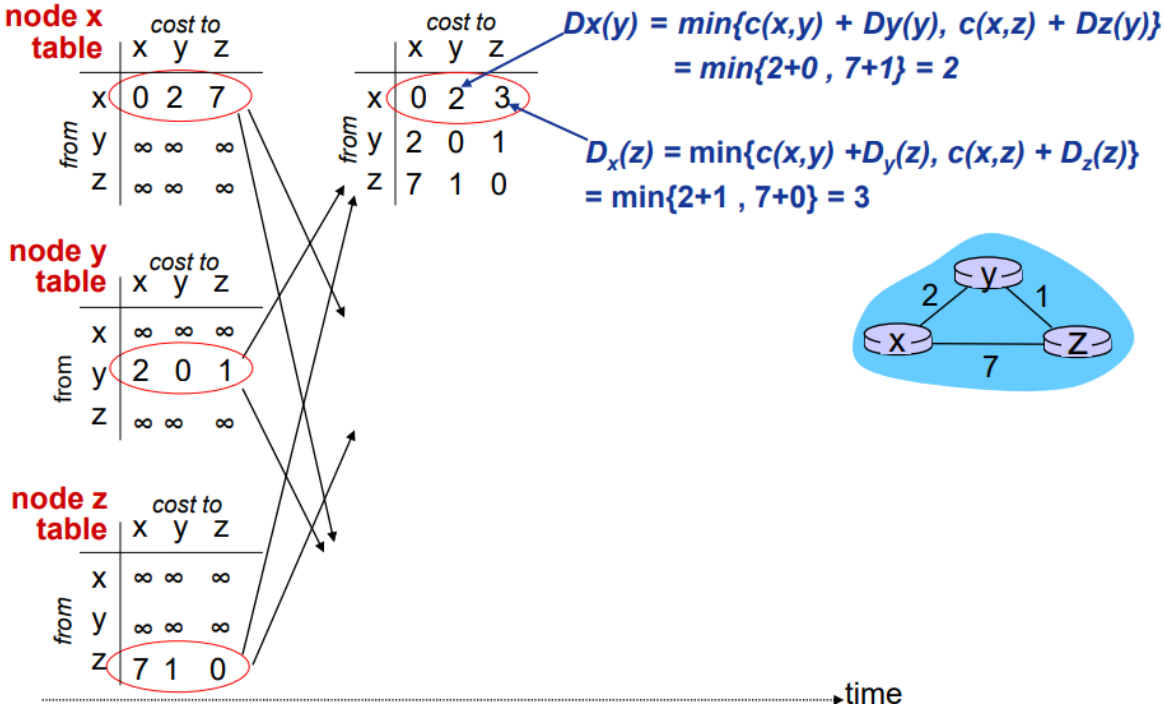
分布式:

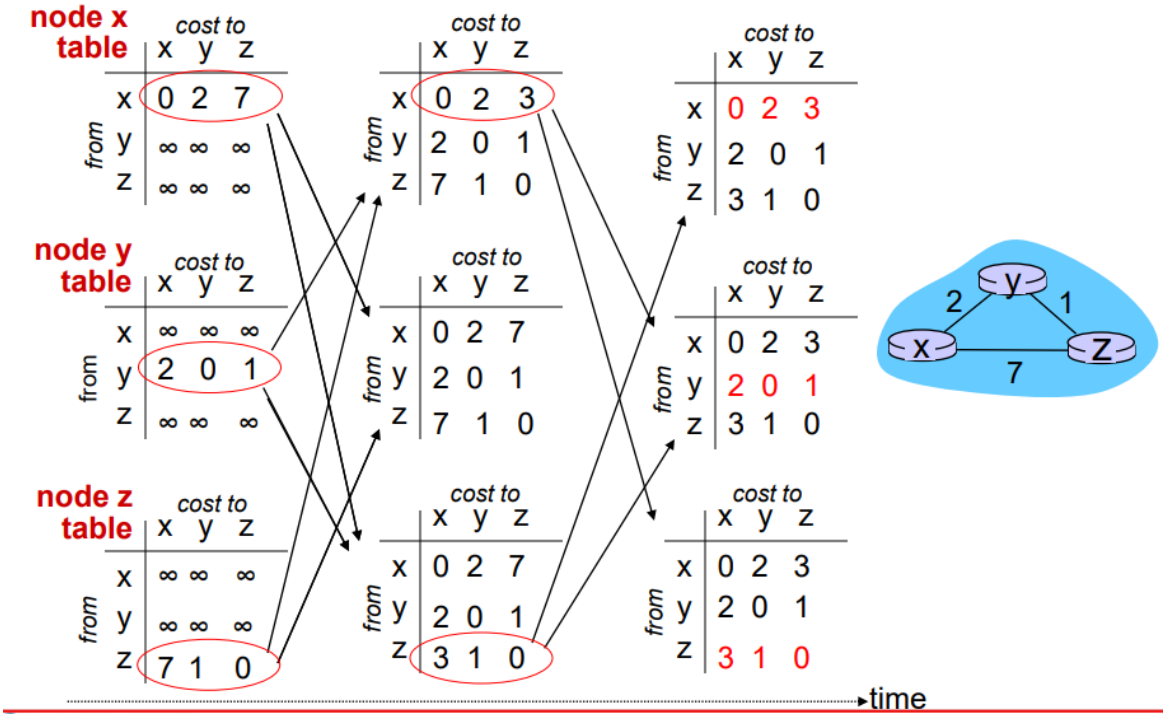
- 每个结点只当DV变化时才通告给邻居
  - 邻居在必要时（其DV更新后发生改变）再通告它们的邻居

每个结点:



距离向量路由算法举例

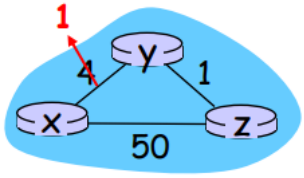




距离向量DV：链路费用变化

链路费用变化:

- 结点检测本地链路费用变化
- 更新路由信息，重新计算距离向量
- 如果DV改变，通告所有邻居

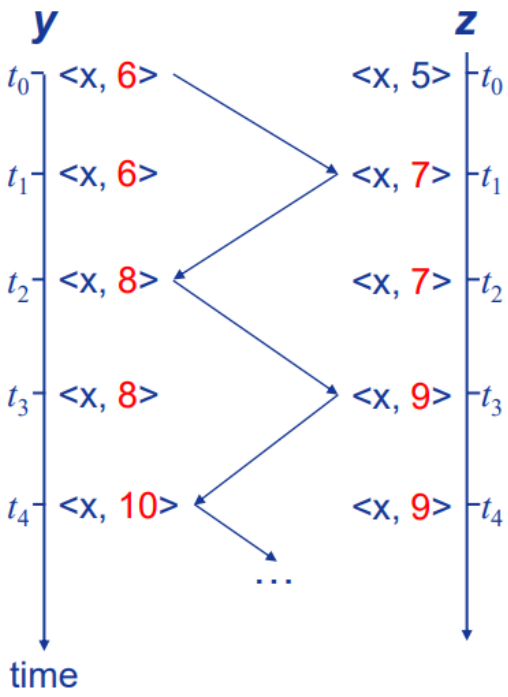


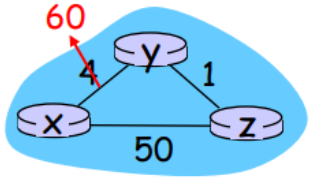
t_0 : y检测到链路费用改变，更新DV，通告其邻居.

t_1 : z收到y的DV更新，更新其距离向量表，计算到达x的最新最小费用，更新其DV，并发送给其所有邻居.

t_2 : y收到z的DV更新，更新其距离向量表，重新计算y的DV，未发生改变，不再向z发送DV.

“☺好消息传播快！”　　“坏消息会怎么样呢？”





坏消息传播慢！

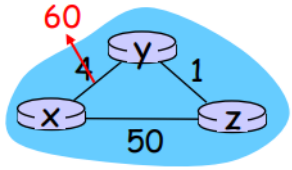
— “无穷计数(count to infinity)”问题！

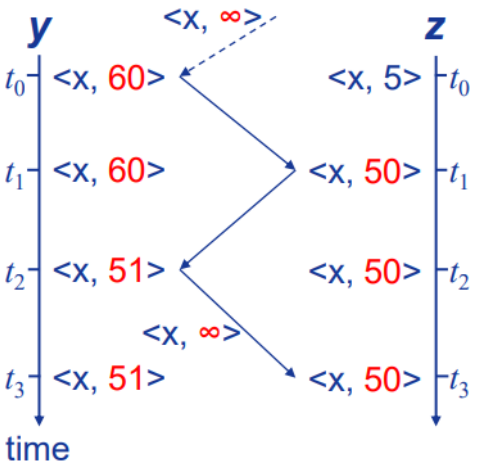
距离向量DV: 无穷计数问题

- 法1

毒性逆转(poisoned reverse):

- 如果一个结点(e.g. Z)到达某目的(e.g.X)的最小费用路径是通过某个邻居(e.g.Y)，则：
  - 通告给该邻居结点到达该目的的距离为无穷大

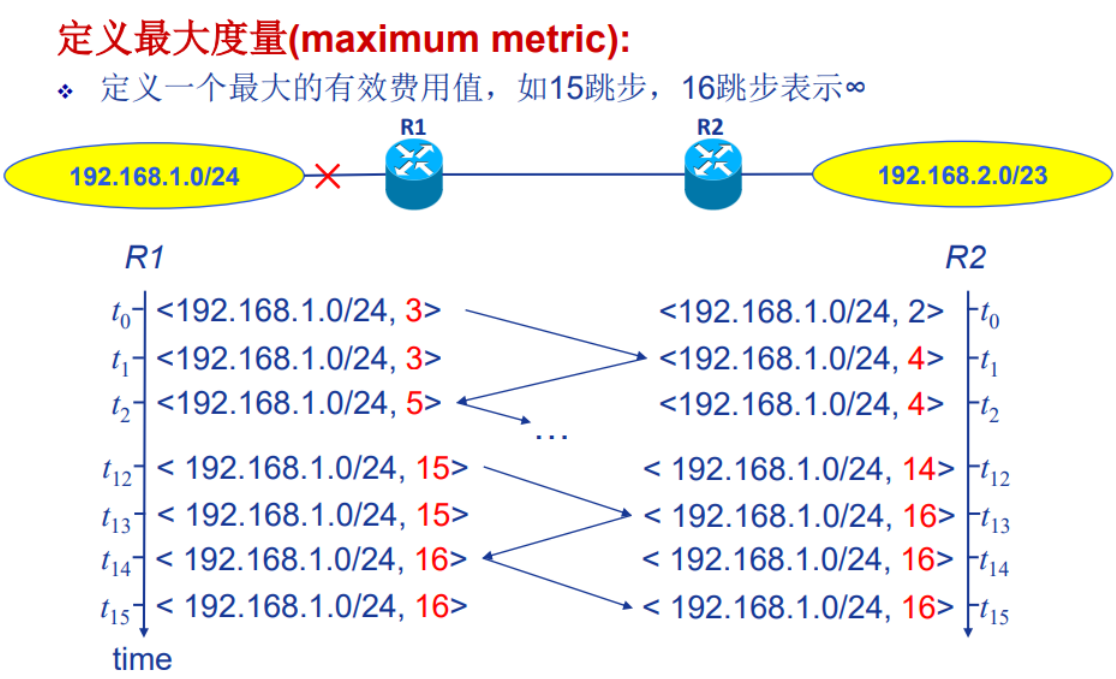




毒性逆转能否彻底解决无穷计数问题？

简单网络可以，更复杂的未必

- 法2



4.9.3 层次化路由

将任意规模网络抽象为一个图计算路由-过于理想化

- 标识所有路由器
- “扁平”网络

——在实际网络（尤其是大规模网络）中，不可行！

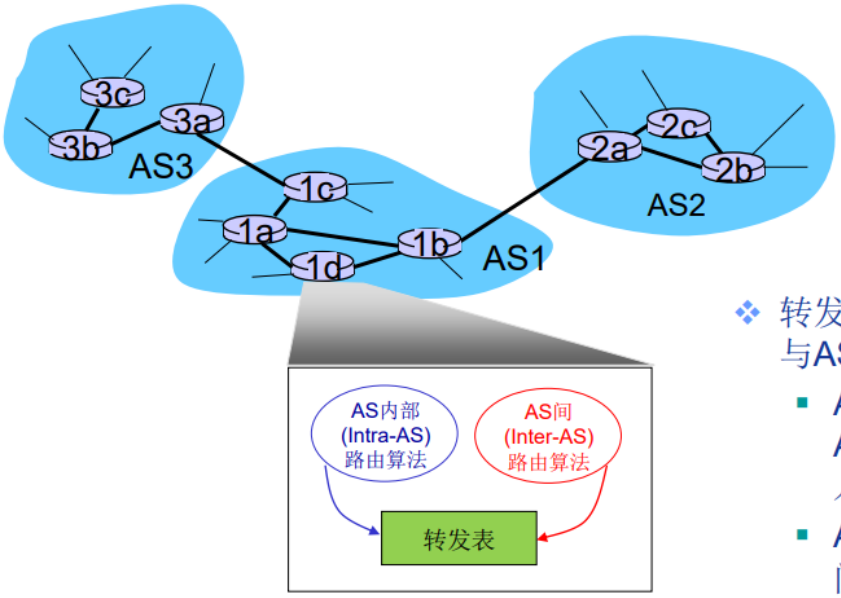
网络规模：考虑6亿目的结点的网络

- 路由表几乎无法存储！
- 路由计算过程的信息（e.g. 链路状态分组、DV）交换量巨大，会淹没链路！

管理自治：

- 每个网络的管理可能都期望自主控制其网内的路由
- 互联网(internet) = 网络之网络(network of networks)

- 聚合路由器为一个区域： 自治系统AS (autonomous systems)
- 同一AS内的路由器运行相同的路由协议(算法)
  - 自治系统**内部路由协议** (“intra-AS” routing protocol)
  - **不同自治系统**内的路由器可以运行**不同**的AS内部路由协议
- 网关路由器(gateway router)
  - 位于AS“边缘”
  - 通过链路连接其他AS的网关路由器
- 例子：互连的AS

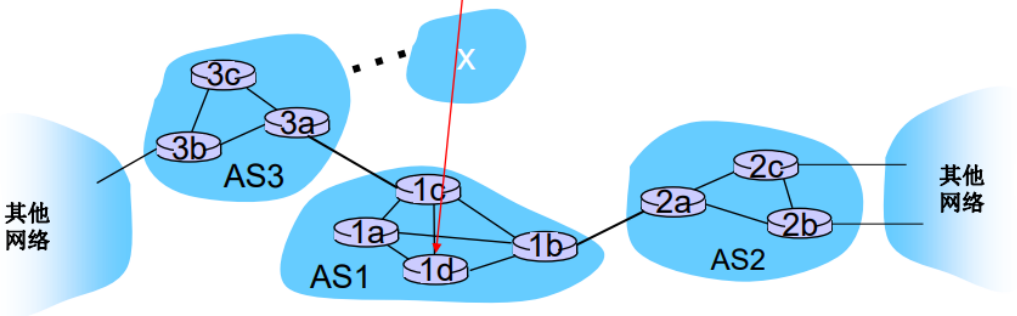


- 转发表由AS内部路由算法与AS间路由算法共同配置
  - AS内部路由算法设置AS内部目的网络路由入口(entries)
  - AS内部路由算法与AS间路由算法共同设置AS外部目的网络路由入口

自治系统间(Inter-AS)路由任务

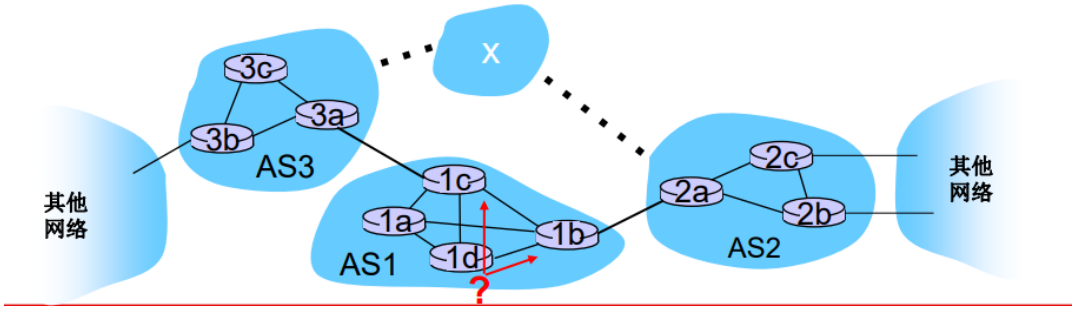
假设AS1内某路由器收到一个目的地址在AS1之外的数据报

- 路由器应该将该数据报转发给哪个网关路由器呢?
  - AS1必须
    1. 学习到哪些目的网络可以通过AS2到达，哪些可以通过AS3到达
    2. 将这些网络可达性信息传播给AS1内部路由器
- 例：路由器1d的转发表设置
  - 假设AS1学习到(通过AS间路由协议)：子网*x*可以通过AS3 (网关 1c)到达，但不能通过AS2到达
    - AS间路由协议向所有内部路由器传播该可达性信息
  - 路由器1d：利用AS内部路由信息，确定其到达1c的最小费用路径接口*I*
    - 在转发表中增加入口：(*x*, *I*)

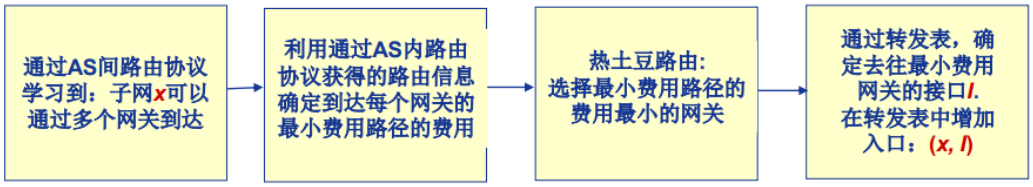


- 例: 在多AS间选择
  -

- 假设AS1通过AS间路由协议学习到：子网*x*通过AS3和AS2均可到达
- 为了配置转发表，路由器1d必须确定应该将去往子网*x*的数据报转发给哪个网关？
  - 这个任务也是由AS间路由协议完成!



- 假设AS1通过AS间路由协议学习到：子网*x*通过AS3和AS2均可到达
- 为了配置转发表，路由器1d必须确定应该将去往子网*x*的数据报转发给哪个网关？
  - 这个任务也是由AS间路由协议完成!
- **热土豆路由: 将分组发送给最近的网关路由器.**



4.10 Internet路由

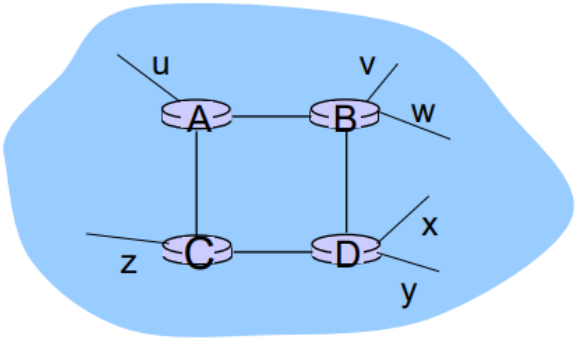
AS内部路由

- Internet采用层次路由
- AS内部路由协议也称为内部网络协议IGP (interior gateway protocols)
- 最常见的AS内部路由协议
  - 路由信息协议：RIP(Routing Information Protocol)
  - 开放最短路径优先：OSPF(Open Shortest Path First)
  - 内部网关路由协议：IGRP(Interior Gateway Routing Protocol)
    - Cisco私有协议

4.10.1 RIP协议

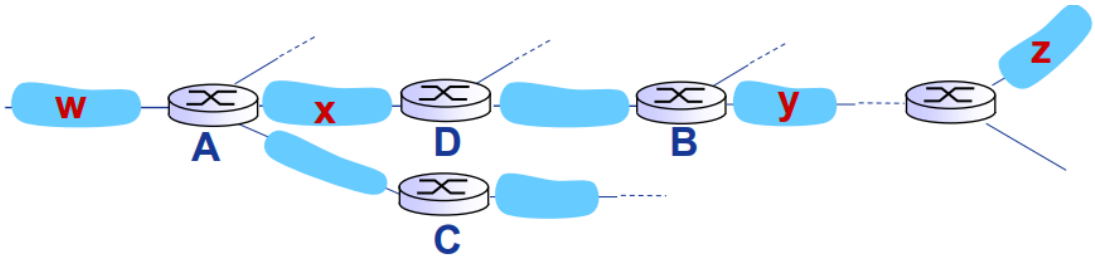
- 早于1982年随BSD-UNIX操作系统发布
- 距离向量路由算法
  - 距离度量：跳步数 (max = 15 hops), 每条链路1个跳步
  - 每隔30秒，邻居之间交换一次DV，成为通告(advertisement)
  - 每次通告：最多25个目的子网(IP地址形式)

从路由器A到目的子网:



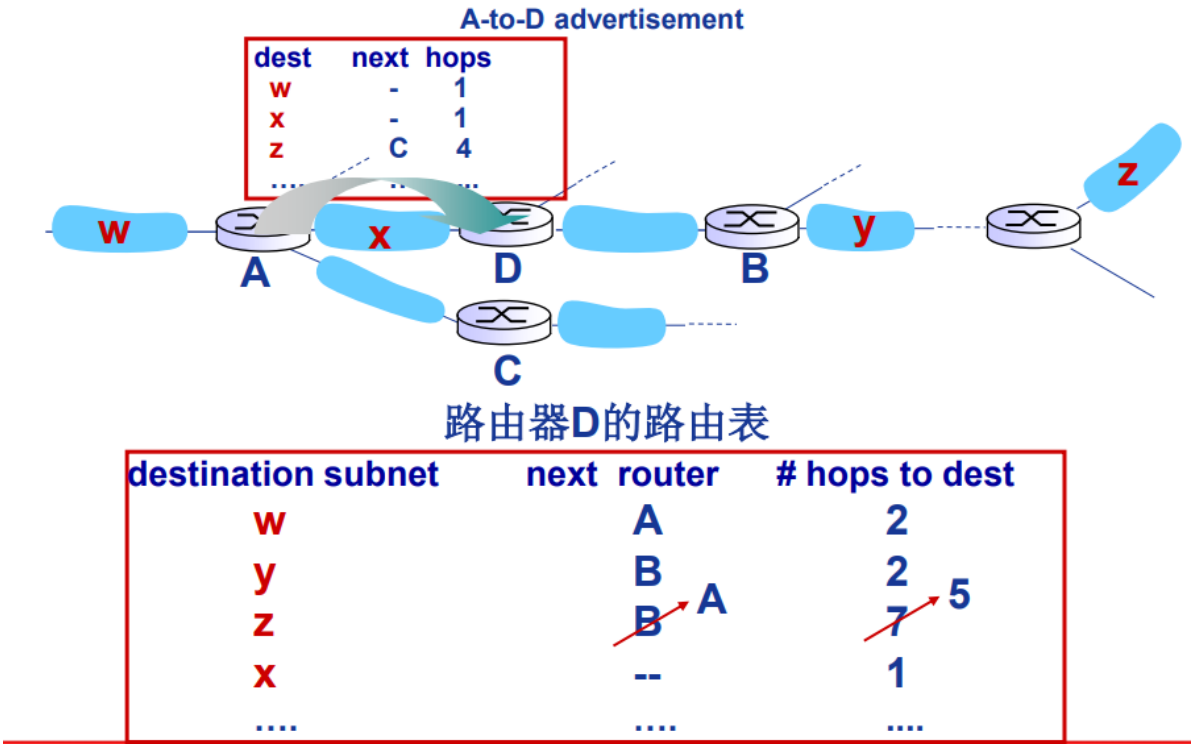
subnet	hops
u	1
v	2
w	2
x	3
y	3
z	2

- 例



路由器D的路由表

destination subnet	next router	# hops to dest
w	A	2
y	B	2
z	B	7
x	--	1
....	....	....



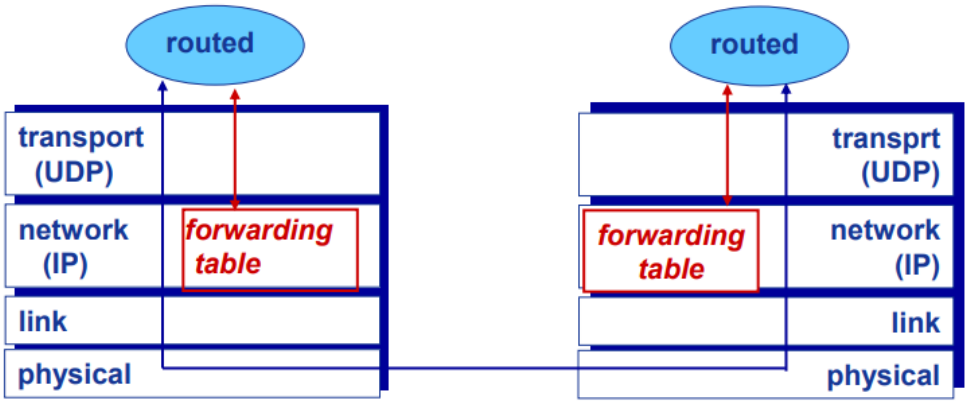
RIP链路失效、恢复

如果180秒没有收到通告→邻居/链路失效

- 经过该邻居的路由不可用
  - 重新计算路由
- 向邻居发送新的通告
- 邻居再依次向外发送通告（如果转发表改变）
- 链路失效信息能否快速传播到全网?
  - 可能发生无穷计数问题
- 毒性逆转技术用于预防乒乓(ping-pong)环路
  - (另外：无穷大距离 = 16 hops)

RIP路由表的处理(网络层功能，但由应用进程实现)

- RIP路由表是利用一个称作route-d (daemon)的应用层进程进行管理
  - 应用进程实现
- 通告报文周期性地通过UDP数据报发送



4.10.2 OSPF协议

OSPF 开放最短路径优先协议

- “开放”: 公众可用
- 采用链路状态路由算法
  - LS分组扩散（通告）
  - 每个路由器构造完整的网络(AS)拓扑图
  - 利用Dijkstra算法计算路由
- OSPF通告中每个入口对应一个邻居
- OSPF通告在整个AS范围泛洪
  - OSPF报文直接封装到IP数据报中
- 与OSPF极其相似的一个路由协议：IS-IS路由协议

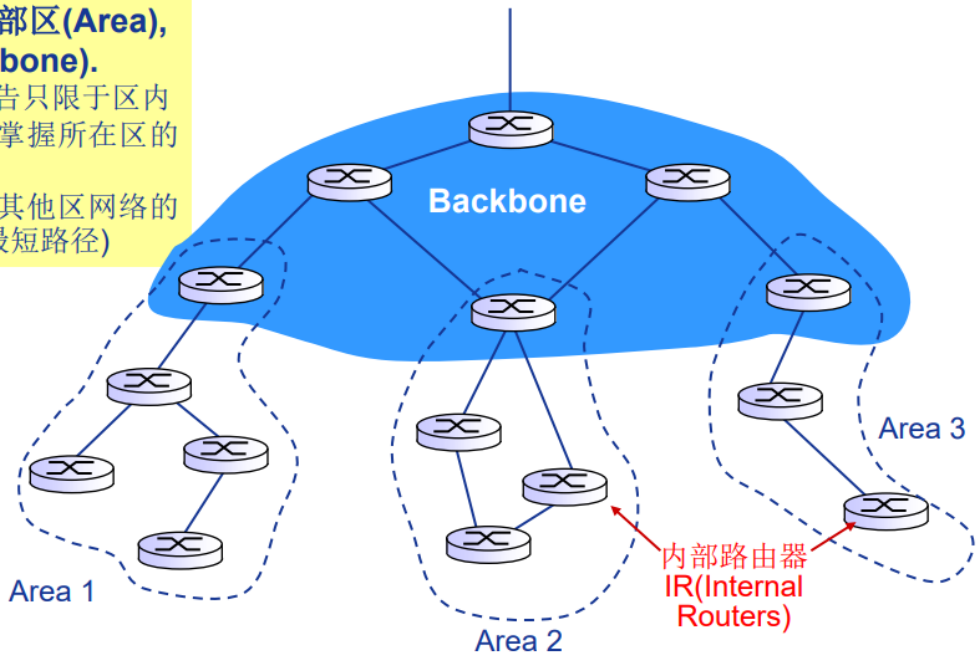
OSPF优点(RIP不具备)

- **安全**(security): 所有OSPF报文可以被认证 (预防恶意入侵)
- 允许使用**多条**相同费用的路径 (RIP只能选一条)
- 对于每条链路，可以针对不同的TOS设置多个不 同的费用度量
  (e.g：卫星链路可以针对“尽力” (best effort) ToS设置“低”费用；针对实时ToS 设置“高”费用)
- 集成单播路由与多播路由
  - 多播OSPF协议(MOSPF) 与OSPF利用相同的网络拓扑数据
- OSPF支持对大规模AS分层(hierarchical)

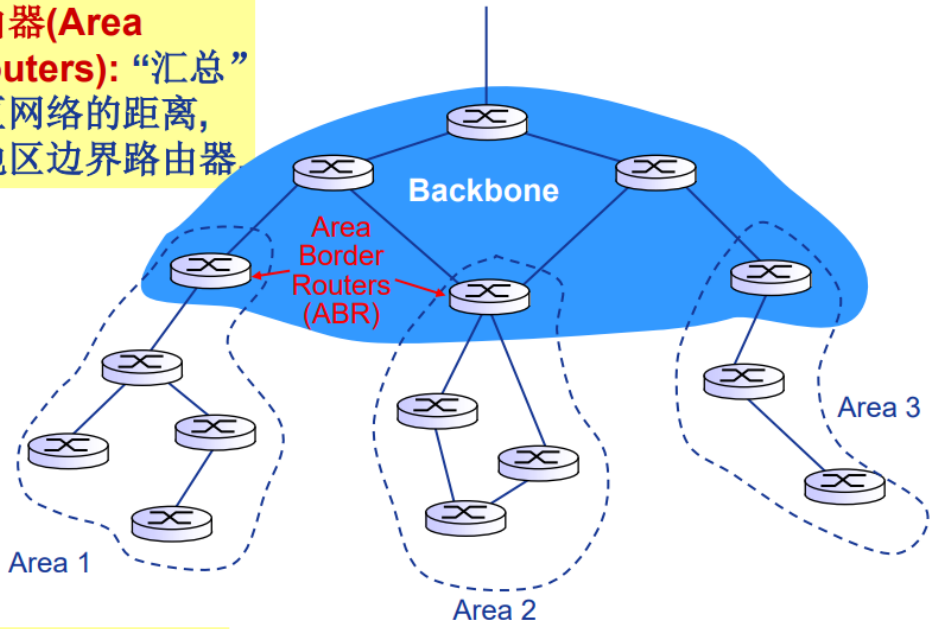
分层的OSPF

两级分层: 局部区(Area), 主干区(Backbone).

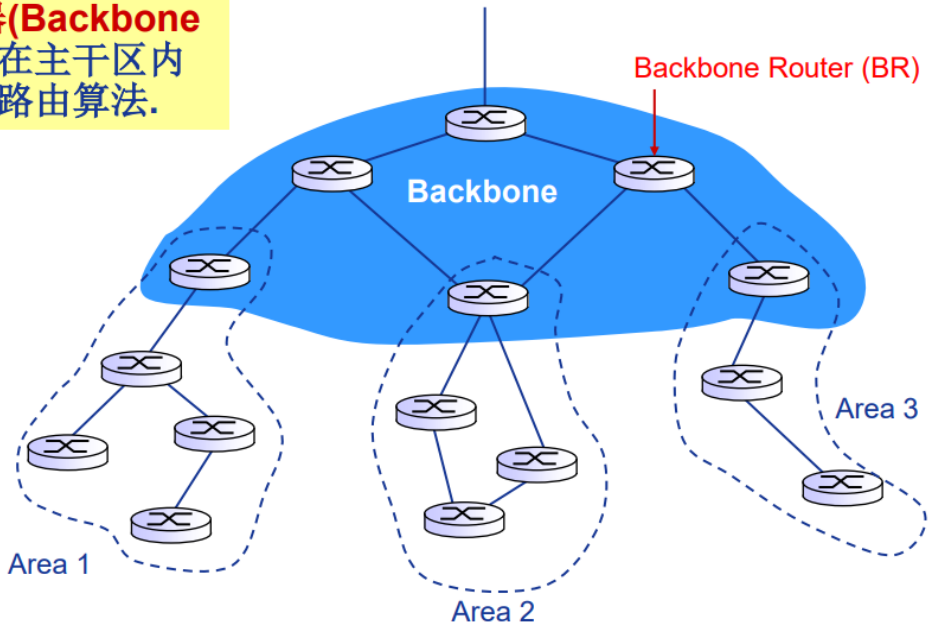
- 链路状态通告只限于区内
- 每个路由器掌握所在区的详细拓扑
- 只知道去往其他区网络的“方向” (最短路径)



区边界路由器(Area Border Routers): “汇总”到达所在区网络的距离，通告给其他区边界路由器.

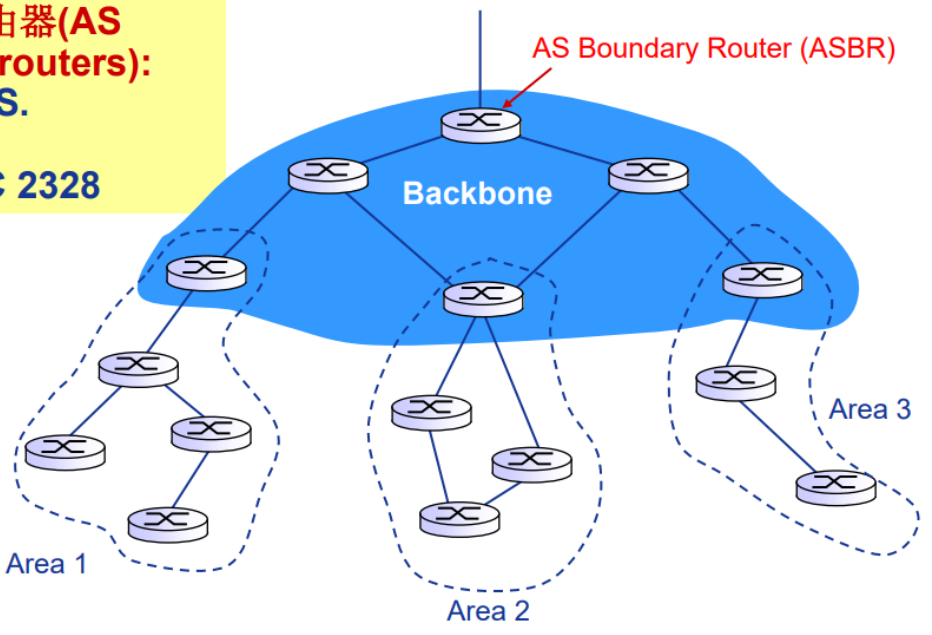


主干路由器(Backbone Routers): 在主干区内运行OSPF路由算法.



AS边界路由器(AS boundary routers): 连接其他AS.

参考：RFC 2328



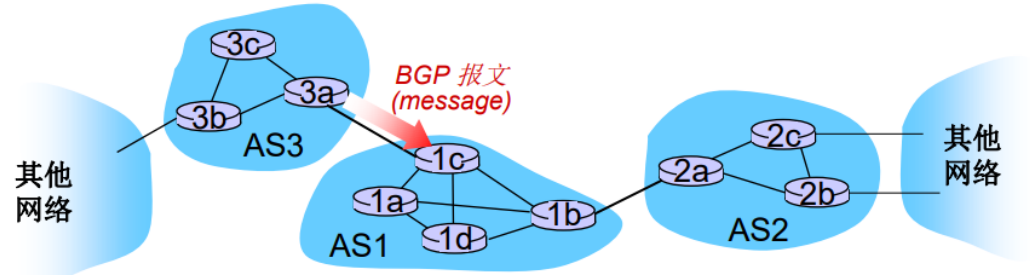
4.10.3 BGP协议

- 边界网关协议BGP (Border Gateway Protocol): 事实上的标准域间路由协议
  - 将Internet “粘合” 为一个整体的关键
- BGP为每个AS提供了一种手段:
  - eBGP: 从邻居AS获取子网可达性信息.
  - iBGP: 向所有AS内部路由器传播子网可达性信息.
  - 基于可达性信息与策略，确定到达其他网络的 “好” 路径.
- 容许子网向Internet其余部分通告它的存在：“我在这儿！”

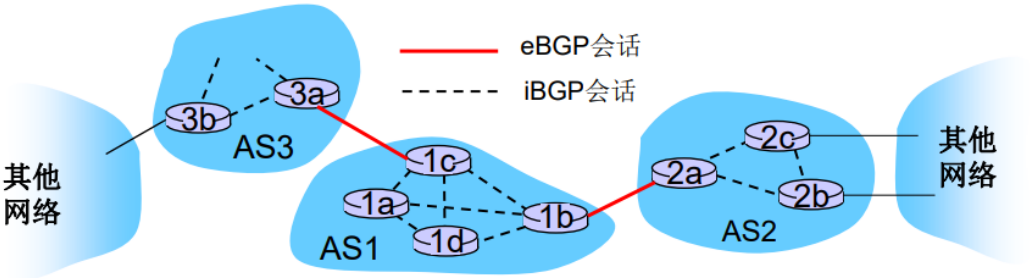
BGP基础

- BGP会话(session): 两个BGP路由器（“Peers”）交换BGP报文:
  - 通告去往不同目的前缀（prefix）的路径 (“路径向量(path vector)” 协议)
  - 报文交换基于半永久的TCP连接
- BGP报文:
  - OPEN: 与peer建立TCP连接，并认证发送方
  - UPDATE: 通告新路径 (或撤销原路径)
  - KEEPALIVE: 在无UPDATE时，保活连接；也用于对OPEN请求的确认
  - NOTIFICATION: 报告先前报文的差错；也被用于关闭连接
-

- 当AS3通告一个前缀给AS1时:
  - AS3承诺可以将数据报转发给该子网
  - AS3在通告中会聚合网络前缀

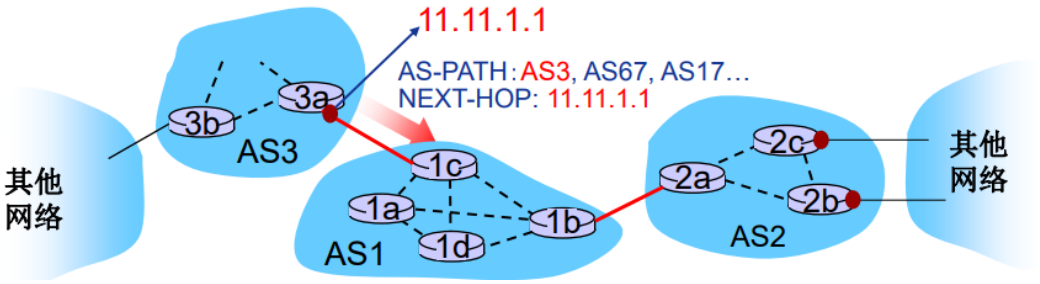


- 分发路径信息
  - 在3a与1c之间, AS3利用eBGP会话向AS1发送前缀可达性信息.
    - 1c则可以利用iBGP向AS1内的所有路由器分发新的前缀可达性信息
    - 1b可以（也可能不）进一步通过1b-到-2a的eBGP会话，向AS2通告新的可达性信息
  - 当路由器获得新的前缀可达性时，即在其转发表中增加关于该前缀的入口（路由项）.



- 路径属性与BGP路由（route）

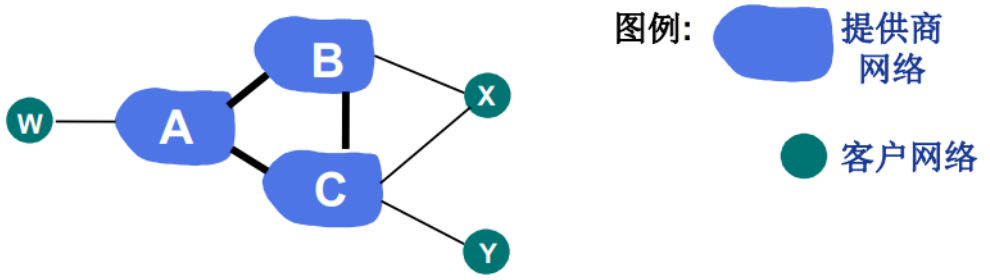
- 通告的前缀信息包括BGP属性
  - 前缀+属性= “路由”
- 两个重要属性:
  - AS-PATH(AS路径): 包含前缀通告所经过的AS序列: e.g., AS 67, AS 17
  - NEXT-HOP(下一跳): 开始一个AS-PATH的路由器接口，指向下一跳AS.
    - 可能从当前AS到下一跳AS存在多条链路



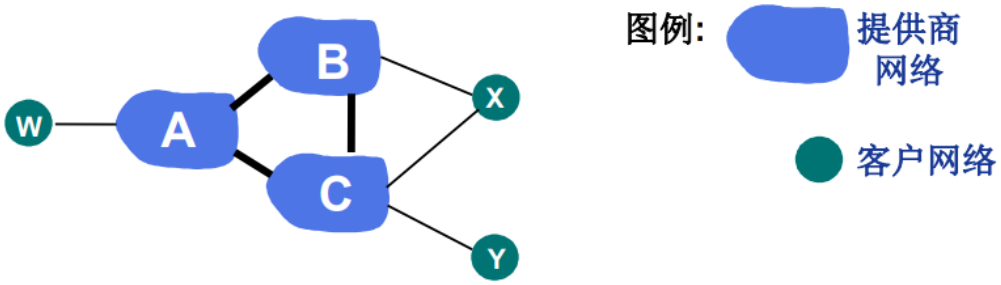
BGP路由选择

- 网关路由器收到路由通告后，利用其输入策略(import policy)决策接受/拒绝该路由
  - e.g., 从不将流量路由到AS x
  - 基于策略(policy-based) 路由
- 路由器可能获知到达某目的AS的多条路由，基于以下准则选择：
  1. 本地偏好(preference)值属性: 策略决策(policy decision)
  2. 最短AS-PATH
  3. 最近NEXT-HOP路由器: 热土豆路由(hot potato routing)
  4. 附加准则

BGP路由选择策略



- A,B,C是提供商网络/AS(provider network/AS)
- X,W,Y是客户网络(customer network/AS)
- W,Y是桩网络(stub network/AS): 只与一个其他AS相连
- X是双宿网络(dual-homed network/AS): 连接两个其他AS
  - X不期望经过他路由B到C的流量
  - … 因此，X不会向B通告任何一条到达C的路由



- A向B通告一条路径：AW
- B向X通告路径：BAW
- B是否应该向C通告路径BAW呢?
  - 绝不! B路由CBAW的流量没有任何“收益”，因为W和C均不是B的客户。
  - B期望强制C通过A向W路由流量
  - B期望只路由去往/来自其客户的流量!

为什么采用不同的AS内与AS间路由协议?

策略(policy):

- inter-AS: 期望能够管理控制流量如何被路由，谁路由经过其网络等.
- intra-AS: 单一管理，无需策略决策

规模(scale):

- 层次路由节省路由表大小，减少路由更新流量
- 适应大规模互联网

性能(performance):

- intra-AS: 侧重性能
- inter-AS: 策略主导