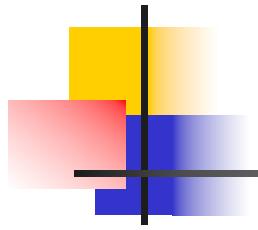


计算机网络

第 4 章 网络层

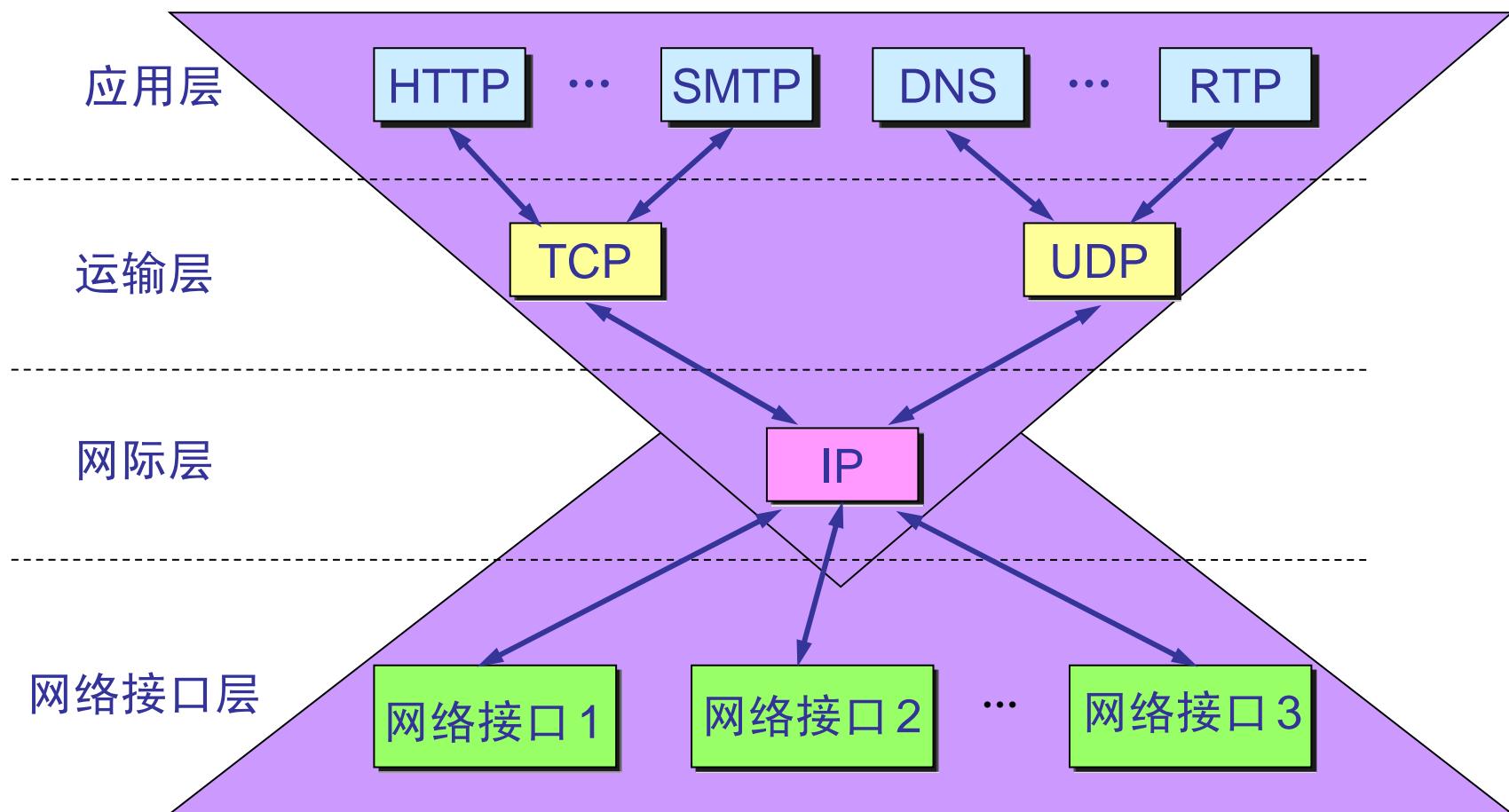
授课老师：洪锋

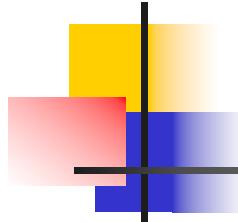
<http://osn.ouc.edu.cn/~hong>



IP over Everything

IP 可应用到各式各样的网络上





第 4 章 网络层

4.1 网络层提供的两种服务

4.2 网际协议 IP

 4.2.1 虚拟互连网络

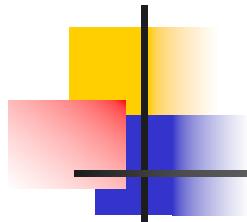
 4.2.2 分类的 IP 地址

 4.2.3 IP 地址与硬件地址

 4.2.4 地址解析协议 ARP 与逆地址解析协议
 RARP

 4.2.5 IP 数据报的格式

 4.2.6 IP 层转发分组的流程



第4章 网络层（续）

4.3 划分子网和构造超网

4.3.1 划分子网

4.3.2 使用子网时分组转发

4.3.3 无分类编址 CIDR（构造超网）

4.4 网际控制报文协议 ICMP

4.4.1 ICMP 报文的种类

4.4.2 ICMP 的应用举例

4.5 因特网的路由选择协议

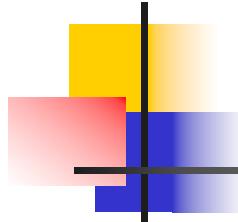
4.5.1 有关路由选择协议的几个基本概念

4.5.2 内部网关协议 RIP

4.5.3 内部网关协议 OSPF

4.5.4 外部网关协议 BGP

4.5.5 路由器的构成



第4章 网络层（续）

4.6 IP 多播

4.6.1 IP 多播的基本概念

4.6.2 在局域网上进行硬件多播

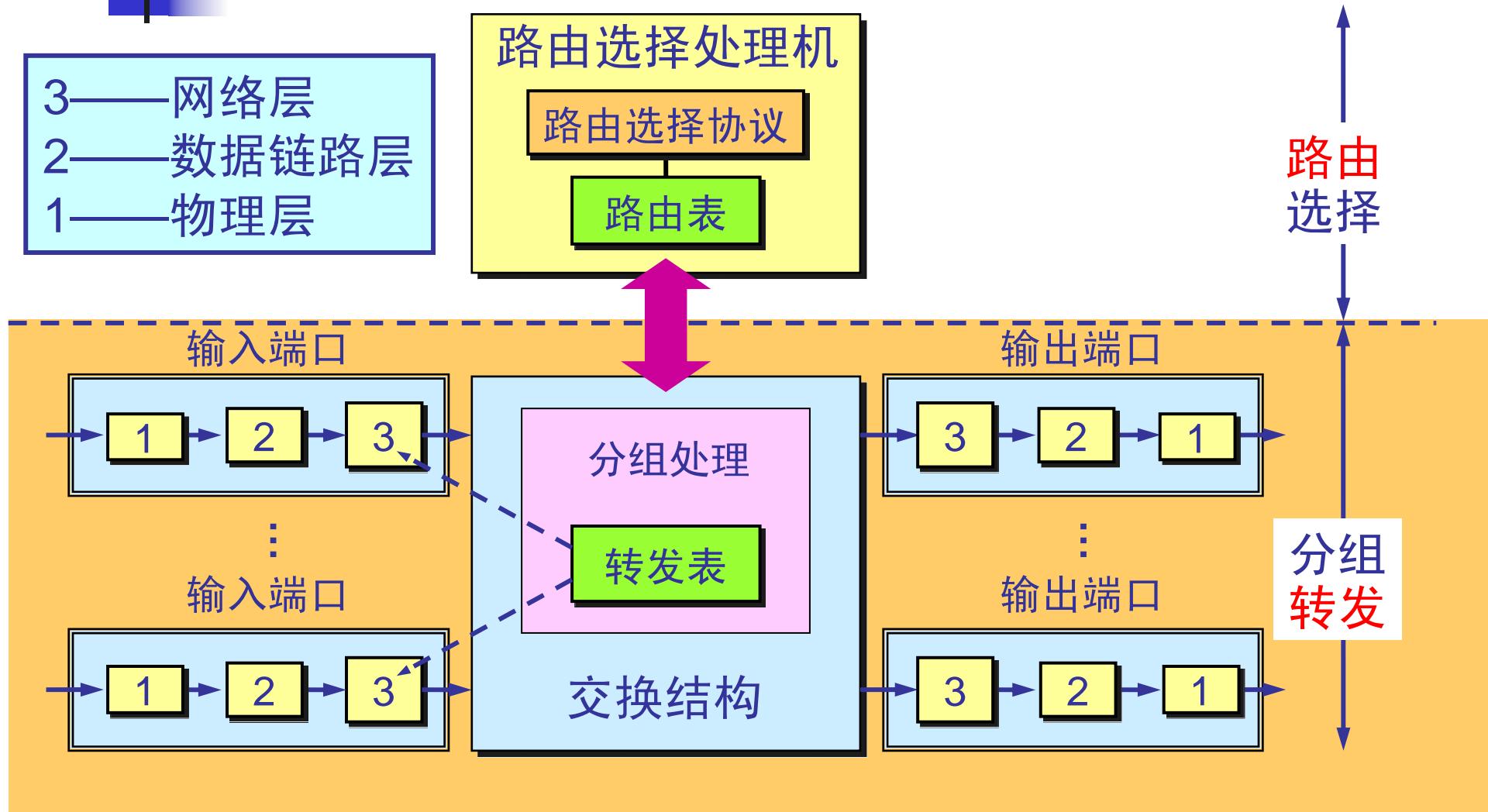
4.6.2 因特网组管理协议 IGMP 和多播路由选择协议

4.7 虚拟专用网 VPN 和网络地址转换 NAT

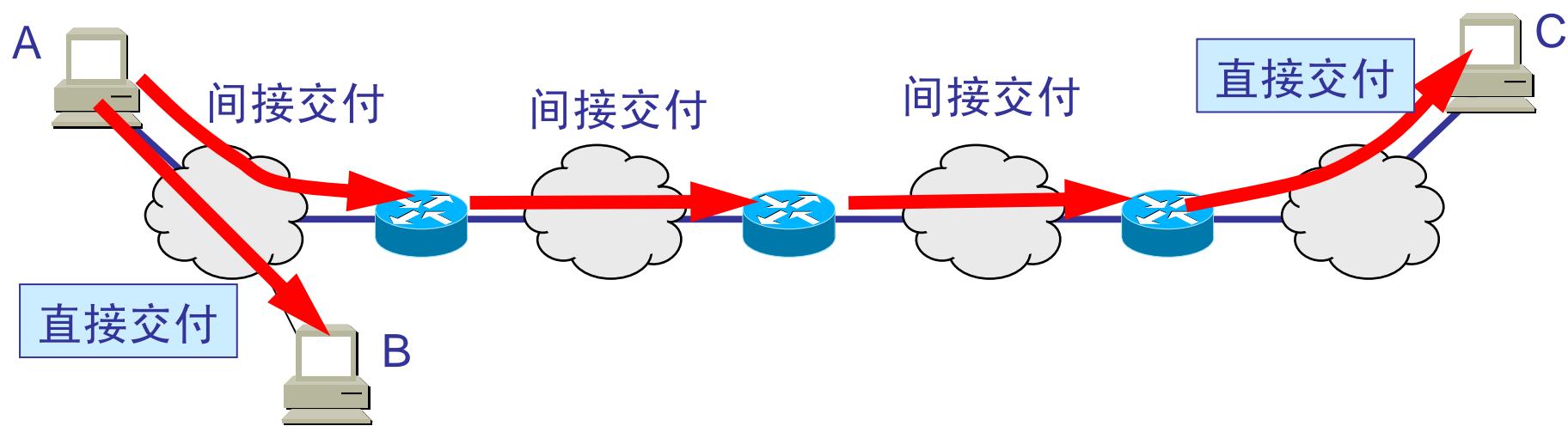
4.7.1 虚拟专用网 VPN

4.7.2 网络地址转换 NAT

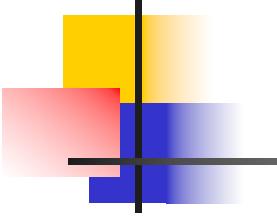
典型的路由器的结构



直接交付和间接交付

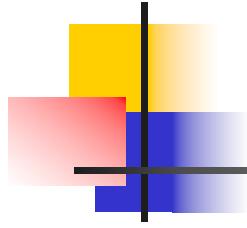


直接交付不需要使用路由器
但间接交付就必须使用路由器



4.1 网络层提供的两种服务

- 在计算机网络领域，网络层应该向运输层提供怎样的服务（“面向连接”还是“无连接”）曾引起了长期的争论。
- 争论焦点的实质就是：在计算机通信中，**可靠交付应当由谁来负责？是网络还是端系统？**



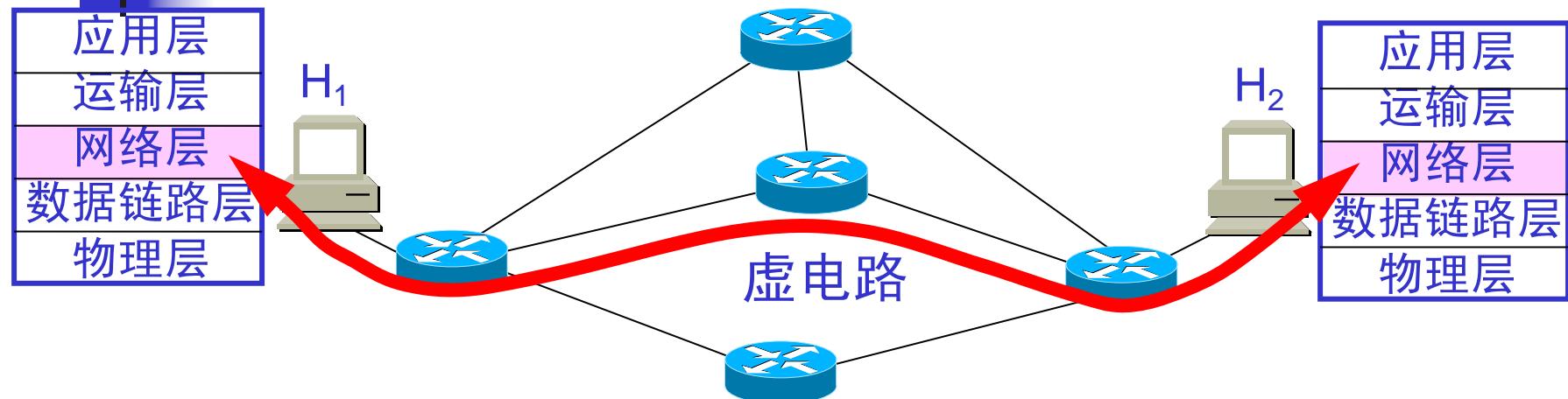
电信网的成功经验

让网络负责可靠交付

■ 面向连接的通信方式

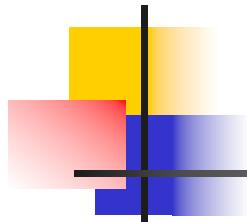
- 建立**虚电路**(Virtual Circuit)，以保证双方通信所需的一切网络资源。
- 如果再使用可靠传输的网络协议，就可使所发送的分组无差错按序到达终点。

虚电路服务



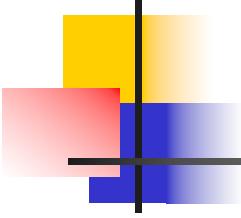
H_1 发送给 H_2 的所有分组都沿着同一条虚电路传送

- 虚电路表示这是一条逻辑上的连接，分组都沿着这条逻辑连接按照存储转发方式传送，而并不是真正建立了一条物理连接。
- 电路交换的电话通信是先建立了一条真正的连接。因此分组交换的虚连接和电路交换的连接只是类似，但并不完全一样。



因特网采用的设计思路

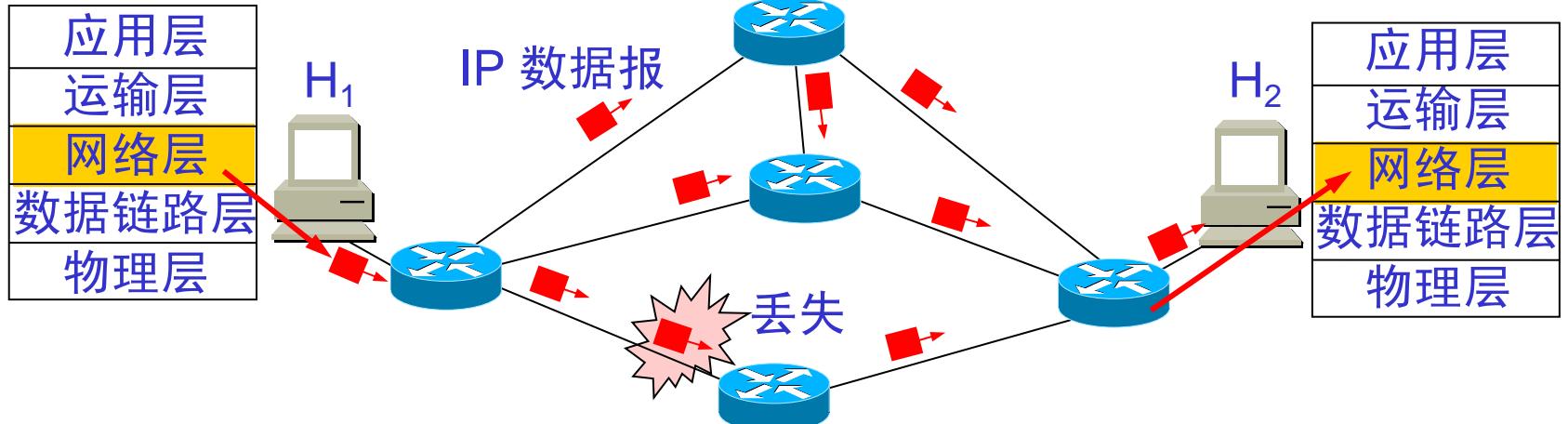
- 网络层向上只提供简单灵活的、**无连接的、尽最大努力交付的数据报服务。**
- 网络在发送分组时不需要先建立连接。每一个分组（即 IP 数据报）独立发送，与其前后的分组无关（不进行编号）。
- 网络层不提供服务质量的承诺。即所传送的分组可能出错、丢失、重复和失序（不按序到达终点），当然也不保证分组传送的时限。



尽最大努力交付的好处

- 由于传输网络不提供端到端的可靠传输服务，这就使网络中的路由器可以做得比较**简单**，而且价格低廉（与电信网的交换机相比较）。
- 如果主机（即端系统）中的进程之间的通信需要是可靠的，那么就由网络的主机中的**运输层**负责（包括差错处理、流量控制等）。
- 采用这种设计思路的好处是：网络的造价大大降低，运行方式灵活，能够适应多种应用。
- 因特网能够发展到今日的规模，充分证明了当初采用这种设计思路的正确性。

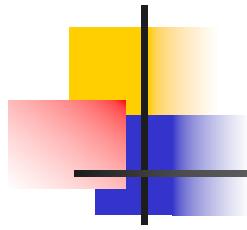
数据报服务



H₁ 发送给 H₂ 的分组可能沿着不同路径传送

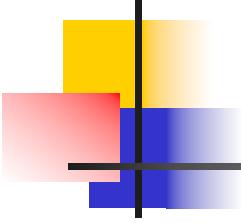
虚电路服务与数据报服务的对比

对比的方面	虚电路服务	数据报服务
思路	可靠通信应当由网络来保证	可靠通信应当由用户主机来保证
连接的建立	必须有	不需要
终点地址	仅在连接建立阶段使用，每个分组使用短的虚电路号	每个分组都有终点的完整地址
分组的转发	属于同一条虚电路的分组均按照同一路由进行转发	每个分组独立选择路由进行转发
当结点出故障时	所有通过出故障的结点的虚电路均不能工作	出故障的结点可能会丢失分组，一些路由可能会发生变化
分组的顺序	总是按发送顺序到达终点	到达终点时不一定按发送顺序
端到端的差错处理和流量控制	可以由网络负责，也可以由用户主机负责	由用户主机负责

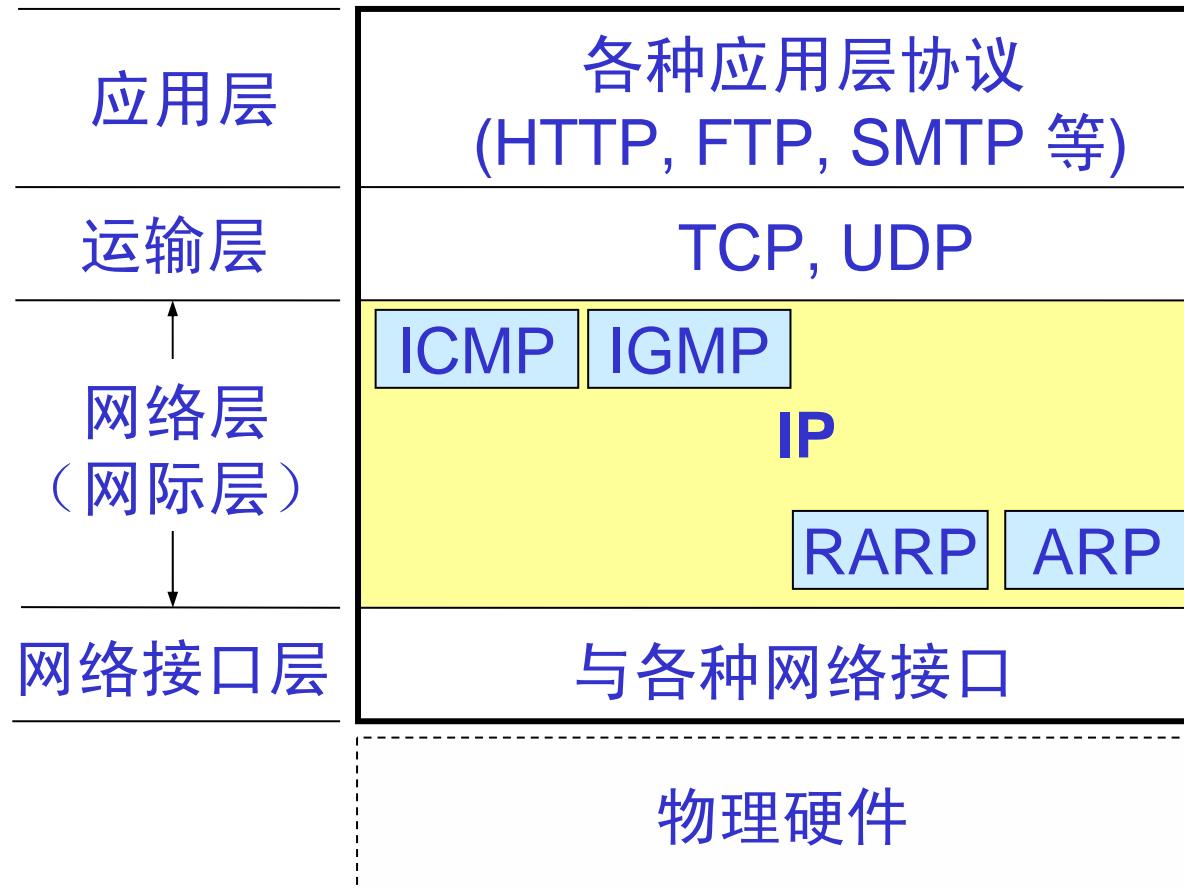


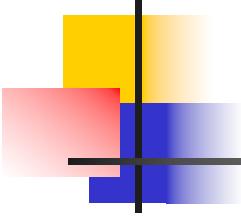
4.2 网际协议IP

- 网际协议 IP 是 TCP/IP 体系中两个最主要的协议之一。
与 IP 协议配套使用的还有四个协议：
 - 地址解析协议 ARP
(Address Resolution Protocol)
 - 逆地址解析协议 RARP
(Reverse Address Resolution Protocol)
 - 网际控制报文协议 ICMP
(Internet Control Message Protocol)
 - 网际组管理协议 IGMP
(Internet Group Management Protocol)



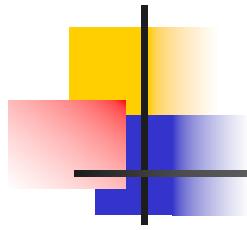
网际层的 IP 协议及配套协议





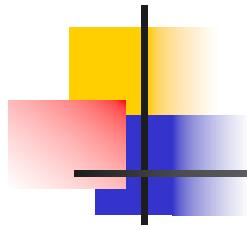
4.2.1 虚拟互连网络

- 互连在一起的网络进行通信，会遇到许多问题：
 - 不同的寻址方案
 - 不同的最大分组长度
 - 不同的网络接入机制
 - 不同的超时控制
 - 不同的差错恢复方法
 - 不同的状态报告方法
 - 不同的路由选择技术
 - 不同的用户接入控制
 - 不同的服务（面向连接服务和无连接服务）
 - 不同的管理与控制方式



网络互相连接起来 要使用中间设备

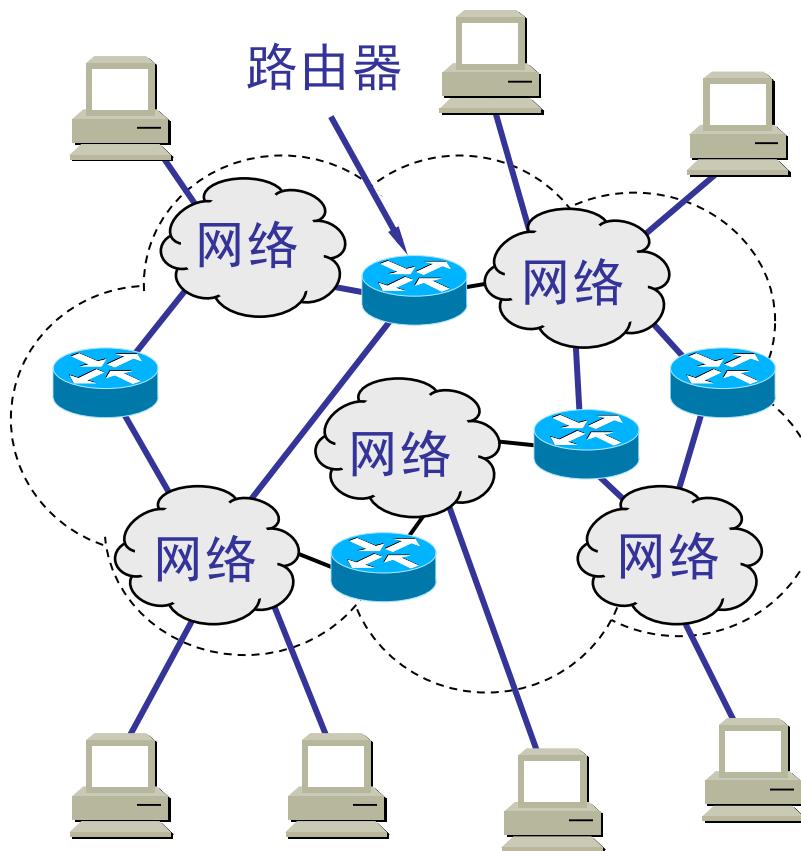
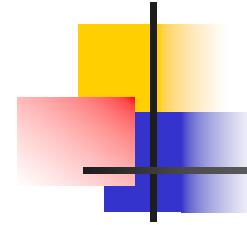
- 中间设备又称为中间系统或中继(relay)系统。
 - 物理层中继系统：转发器(repeater)。
 - 数据链路层中继系统：网桥或桥接器(bridge)。
 - 网络层中继系统：路由器(router)。
 - 网桥和路由器的混合物：桥路器(brouter)。
 - 网络层以上的中继系统：网关(gateway)。



网络互连使用路由器

- 当中继系统是转发器或网桥时，一般并不称之为网络互连，因为这仅仅是把一个网络扩大了，而这仍然是一个网络。
- 网关由于比较复杂，目前使用得较少。
- 互联网都是指用路由器进行互连的网络。
- 由于历史的原因，许多有关 TCP/IP 的文献将网络层使用的路由器称为**网关**。

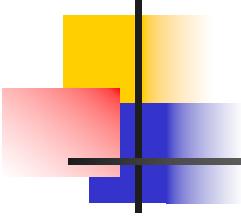
互连网络与虚拟互连网络



(a) 互连网络



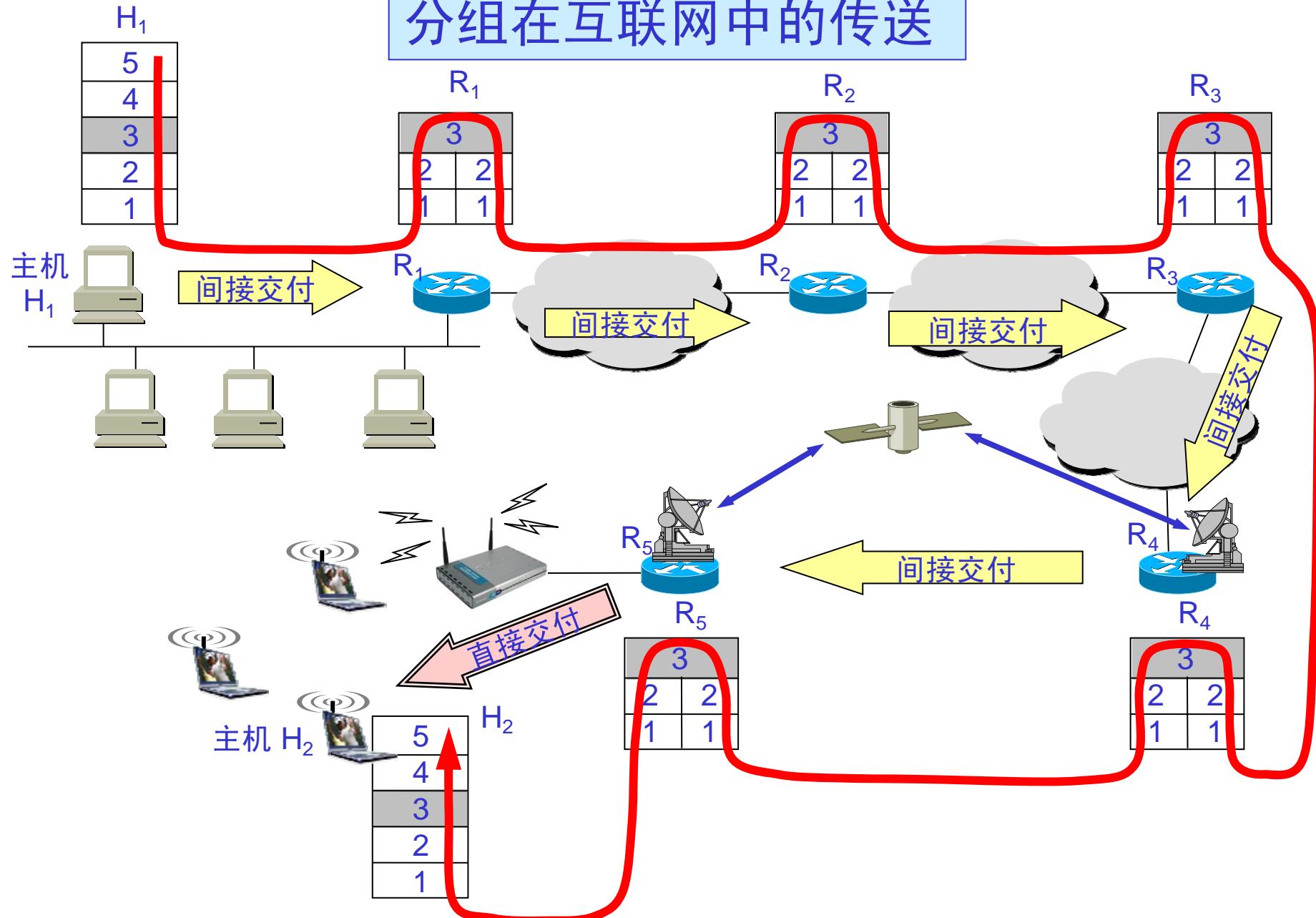
(b) 虚拟互连网络

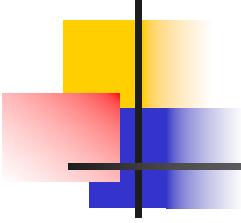


虚拟互连网络的意义

- 所谓虚拟互连网络也就是逻辑互连网络：
 - 互连起来的各种物理网络的异构性本来是客观存在的，但是利用 IP 协议就可以使这些性能各异的网络从用户看起来好像是一个统一的网络。
- 使用 IP 协议的虚拟互连网络可简称为 IP 网。
- 使用虚拟互连网络的好处：
 - 当互联网上的主机进行通信时，就好像在一个网络上通信一样，而看不见互连的各具体的网络异构细节。

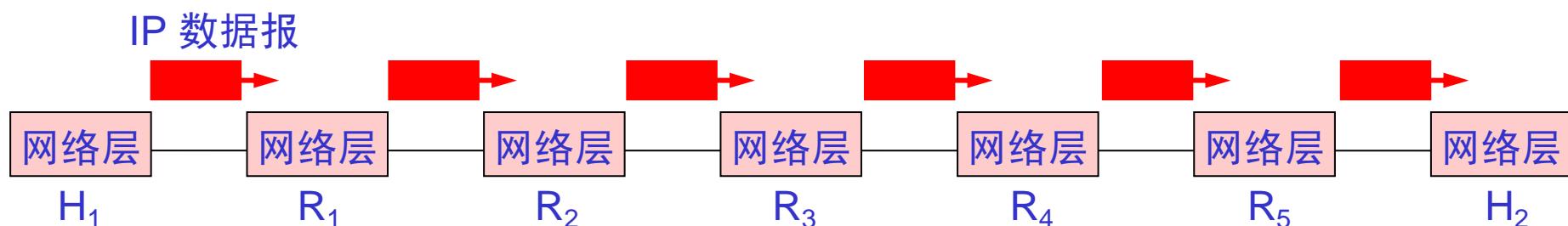
分组在互联网中的传送

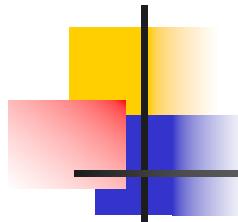




从网络层看 IP 数据报的传送

- 如果我们只从网络层考虑问题，那么 IP 数据报就可以想象是在网络层中传送。

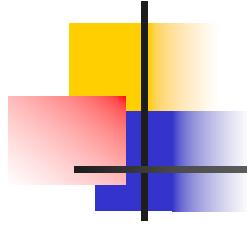




4.2.2 分类的 IP 地址

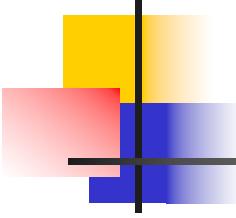
1. IP 地址及其表示方法

- 我们把整个因特网看成为一个单一的、抽象的网络。IP 地址就是给每个连接在因特网上的主机（或路由器）分配一个在全世界范围是唯一的 32 位的标识符。
- IP 地址现在由**因特网名字与号码指派公司 ICANN (Internet Corporation for Assigned Names and Numbers)**进行分配。



IP 地址的编址方法

- 分类的 IP 地址。这是最基本的编址方法，在 1981 年就通过了相应的标准协议。
- 子网的划分。这是对最基本的编址方法的改进，其标准[RFC 950]在 1985 年通过。
- 构成超网。这是比较新的无分类编址方法。1993 年提出后很快就得到推广应用。



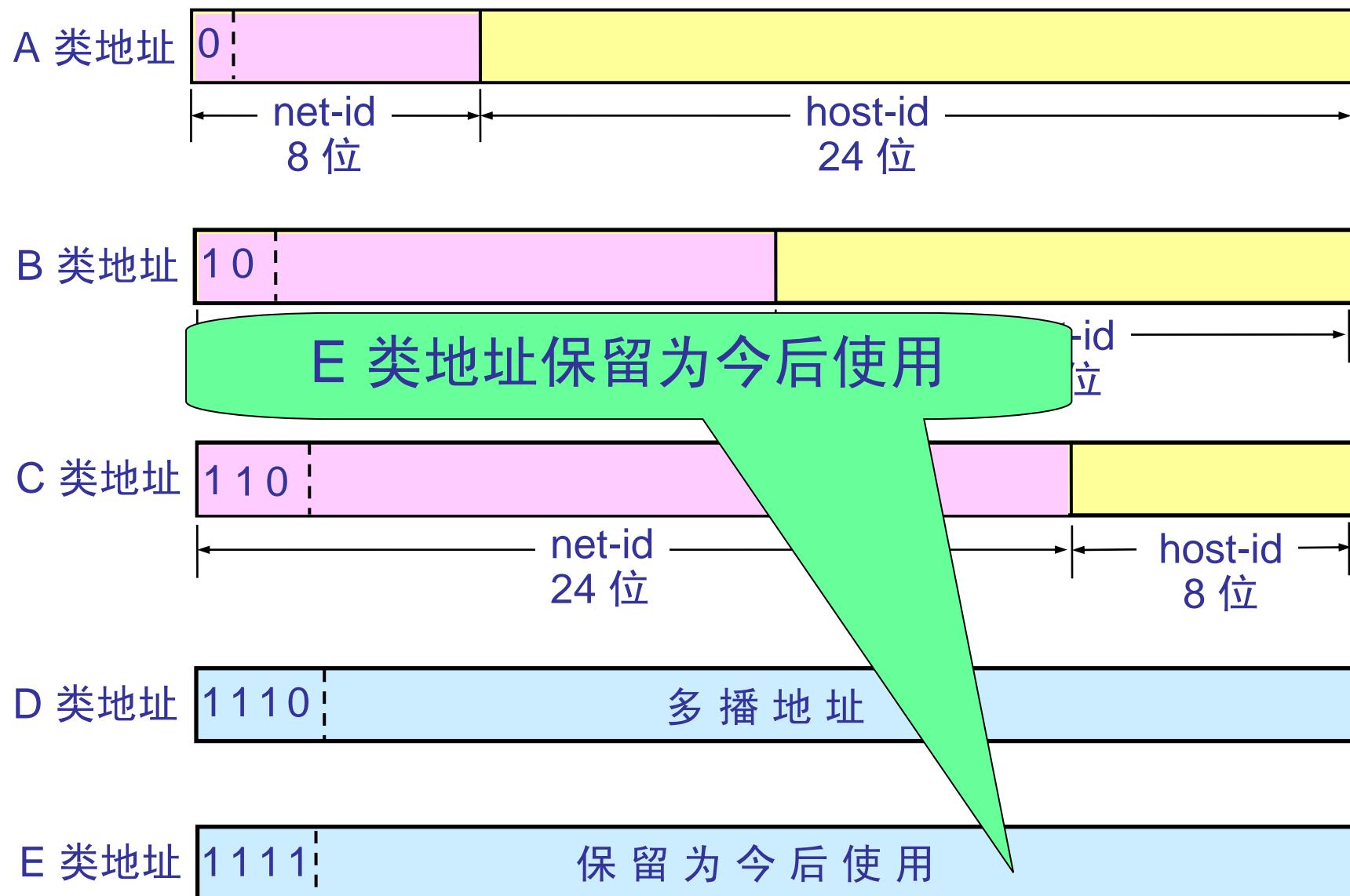
分类 IP 地址

- 每一类地址都由两个固定长度的字段组成，其中一个字段是**网络号 net-id**，它标志主机（或路由器）所连接到的网络，而另一个字段则是**主机号 host-id**，它标志该主机（或路由器）。
- 两级的 IP 地址可以记为：

IP 地址 ::= { <网络号>, <主机号>} (4-1)

::= 代表“定义为”

IP 地址中的网络号字段和主机号字段



点分十进制记法

机器中存放的 IP 地址
是 32 位二进制代码

1000000000001011000000110001111

每隔 8 位插入一个空格
能够提高可读性

10000000 00001011 00000011 00011111

将每 8 位的二进制数
转换为十进制数

128

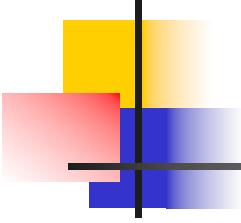
11

3

31

采用点分十进制记法
则进一步提高可读性

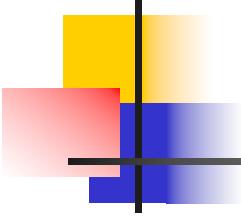
128.11.3.31



2. 常用的三种类别的 IP 地址

IP 地址的使用范围

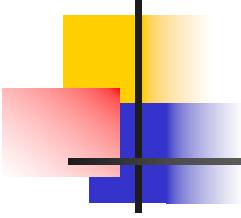
网络类别	最大网络数	第一个可用的网络号	最后一个可用的网络号	每个网络中最大的主机数
A	$126 (2^7 - 2)$	1	126	16,777,214
B	$16,383(2^{14} - 1)$	128.1	191.255	65,534
C	$2,097,151 (2^{21} - 1)$	192.0.1	223.255.255	254



IP 地址的一些重要特点

(1) IP 地址是一种分等级的地址结构。分两个等级的好处：

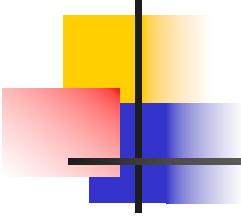
- IP 地址管理机构在分配 IP 地址时只分配网络号，而剩下的主机号则由得到该网络号的单位自行分配。这样就方便了 IP 地址的管理。
- 路由器仅根据目的主机所连接的网络号来转发分组（而不考虑目的主机号），这样就可以使路由表中的项目数大幅度减少，从而减小了路由表所占的存储空间。



IP 地址的一些重要特点

(2) 实际上 IP 地址是标志一个主机（或路由器）和一条链路的接口。

- 当一个主机同时连接到两个网络上时，该主机就必须同时具有两个相应的 IP 地址，其网络号 net-id 必须是不同的。这种主机称为多归属主机(multihomed host)。
- 由于一个路由器至少应当连接到两个网络（这样它才能将 IP 数据报从一个网络转发到另一个网络），因此一个路由器至少应当有两个不同的 IP 地址。

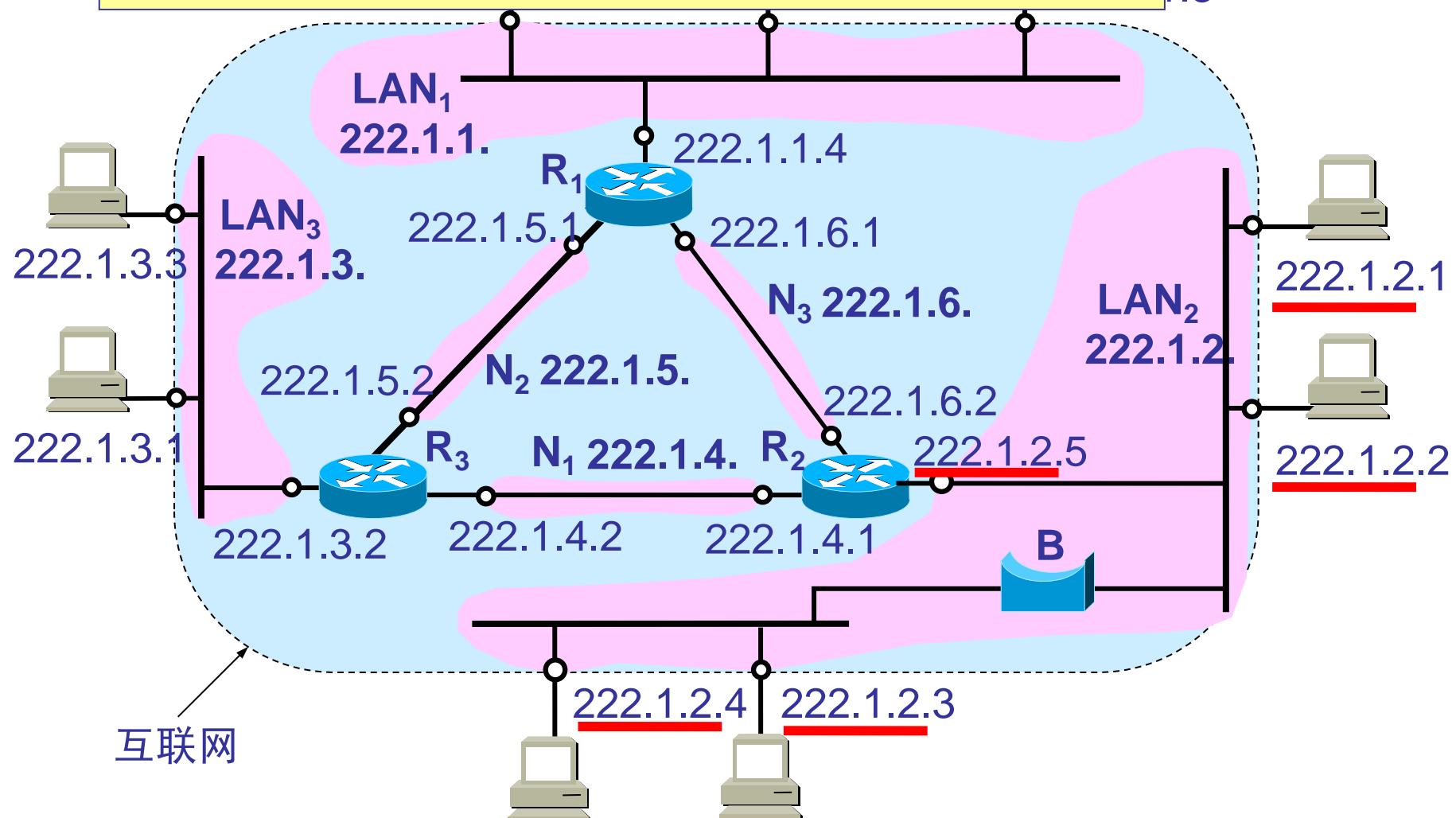


IP 地址的一些重要特点

- (3) 用转发器或网桥连接起来的若干个局域网仍为一个网络，因此这些局域网都具有同样的网络号 net-id。
- (4) 所有分配到网络号 net-id 的网络，范围很小的局域网，还是可能覆盖很大地理范围的广域网，都是平等的。

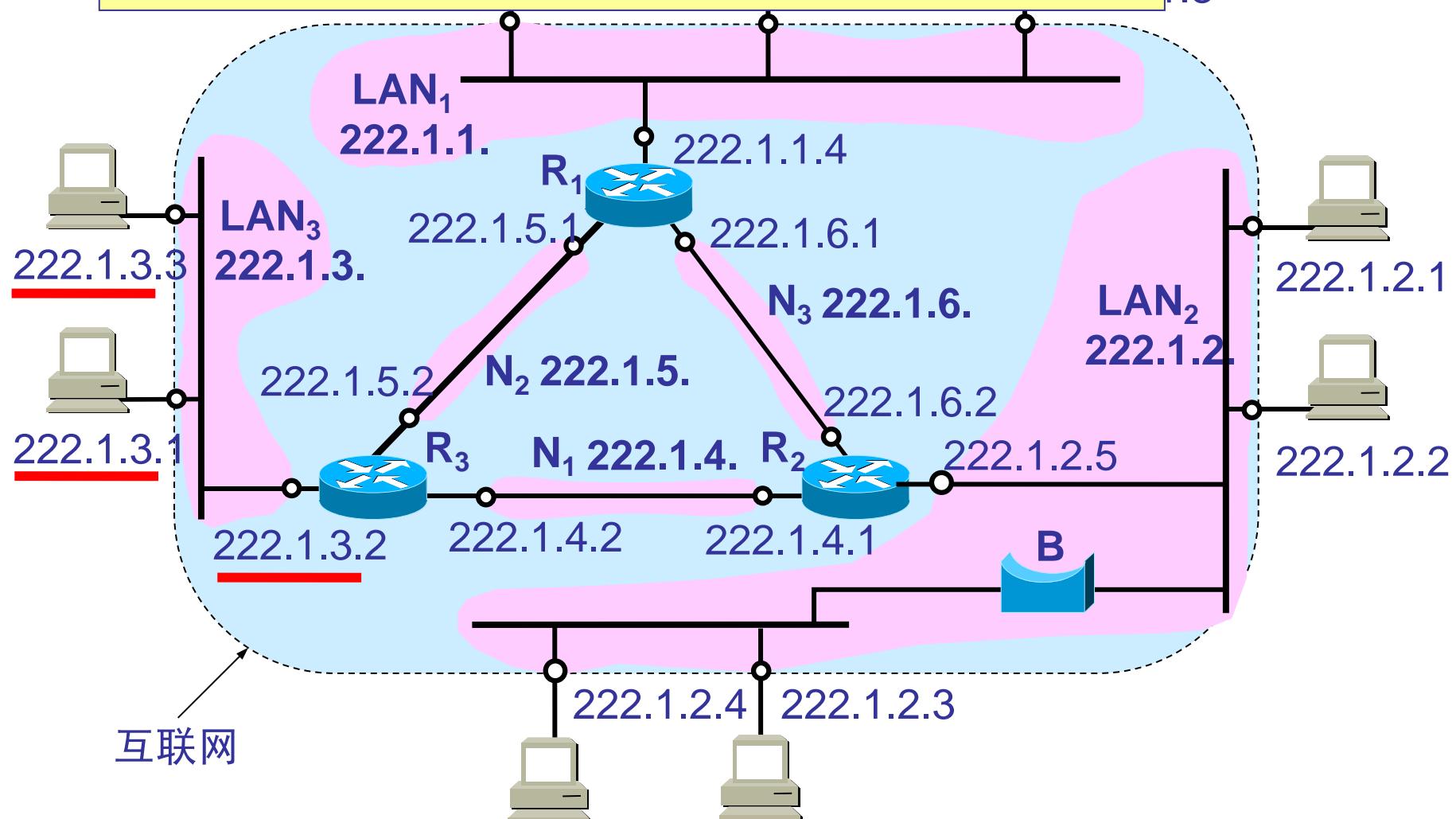
在同一个局域网上的主机或路由器的
IP 地址中的网络号必须是一样的。

图中的网络号就是 IP 地址中的 net-id

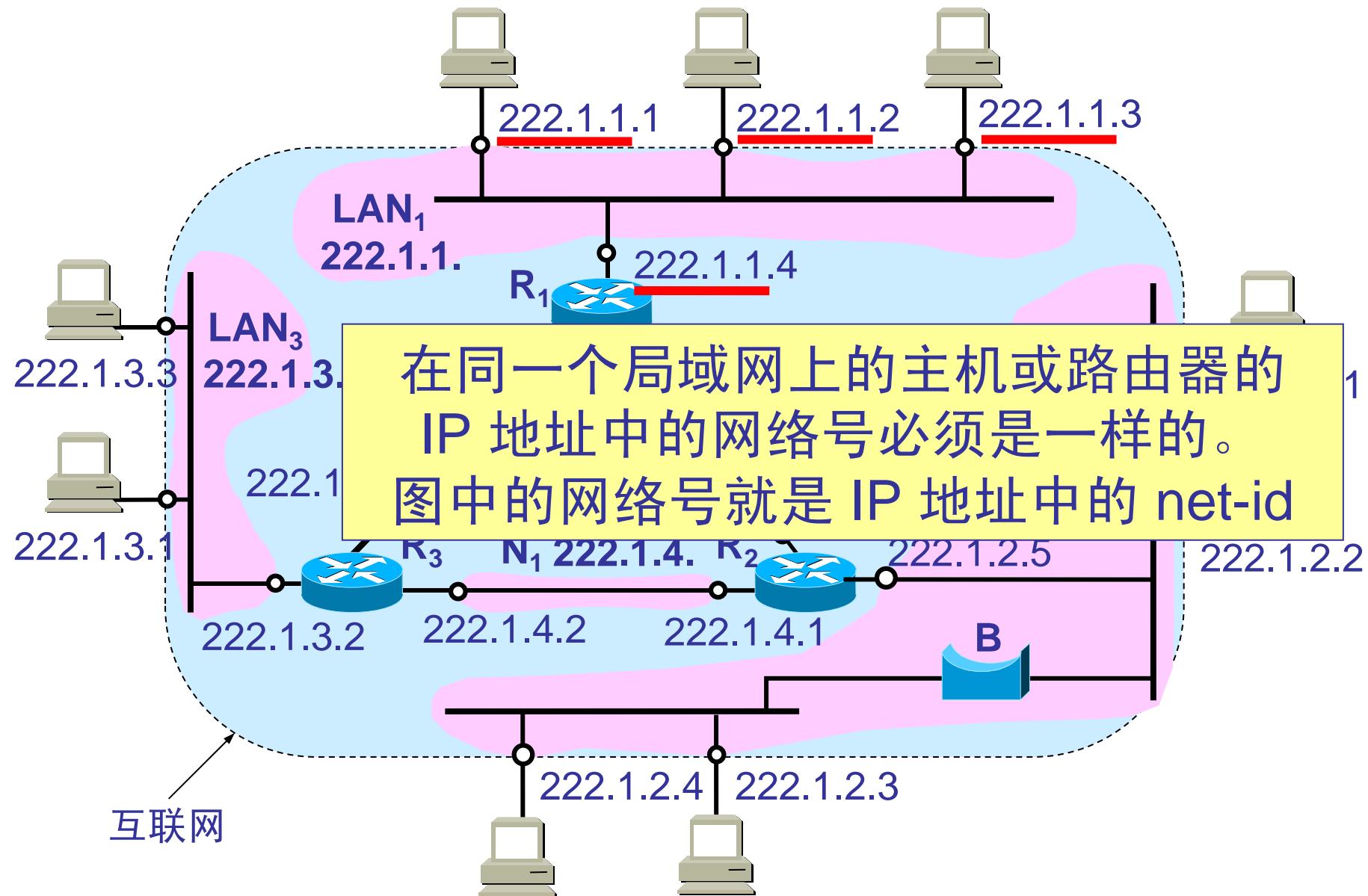


在同一个局域网上的主机或路由器的
IP 地址中的网络号必须是一样的。

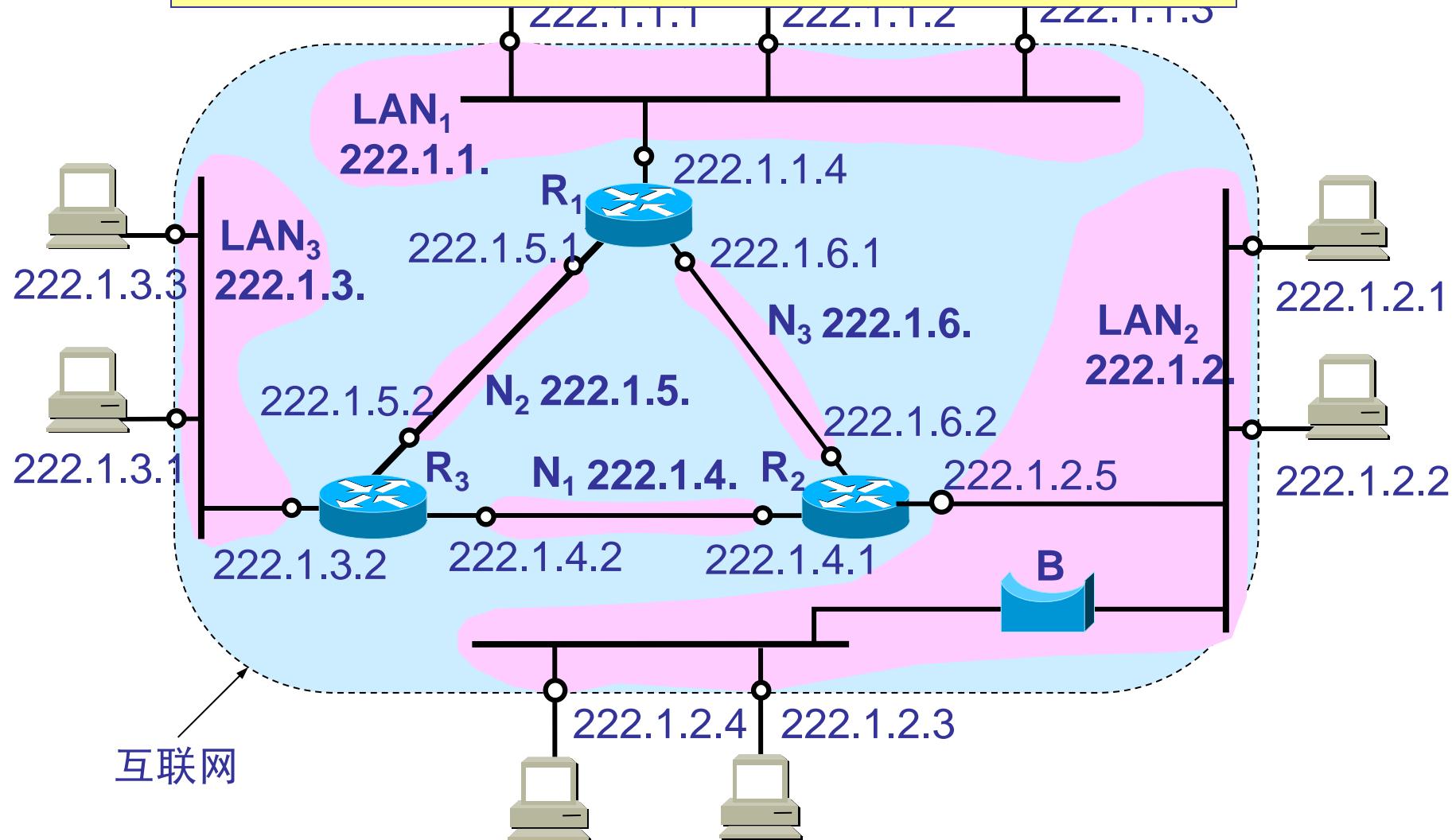
图中的网络号就是 IP 地址中的 net-id



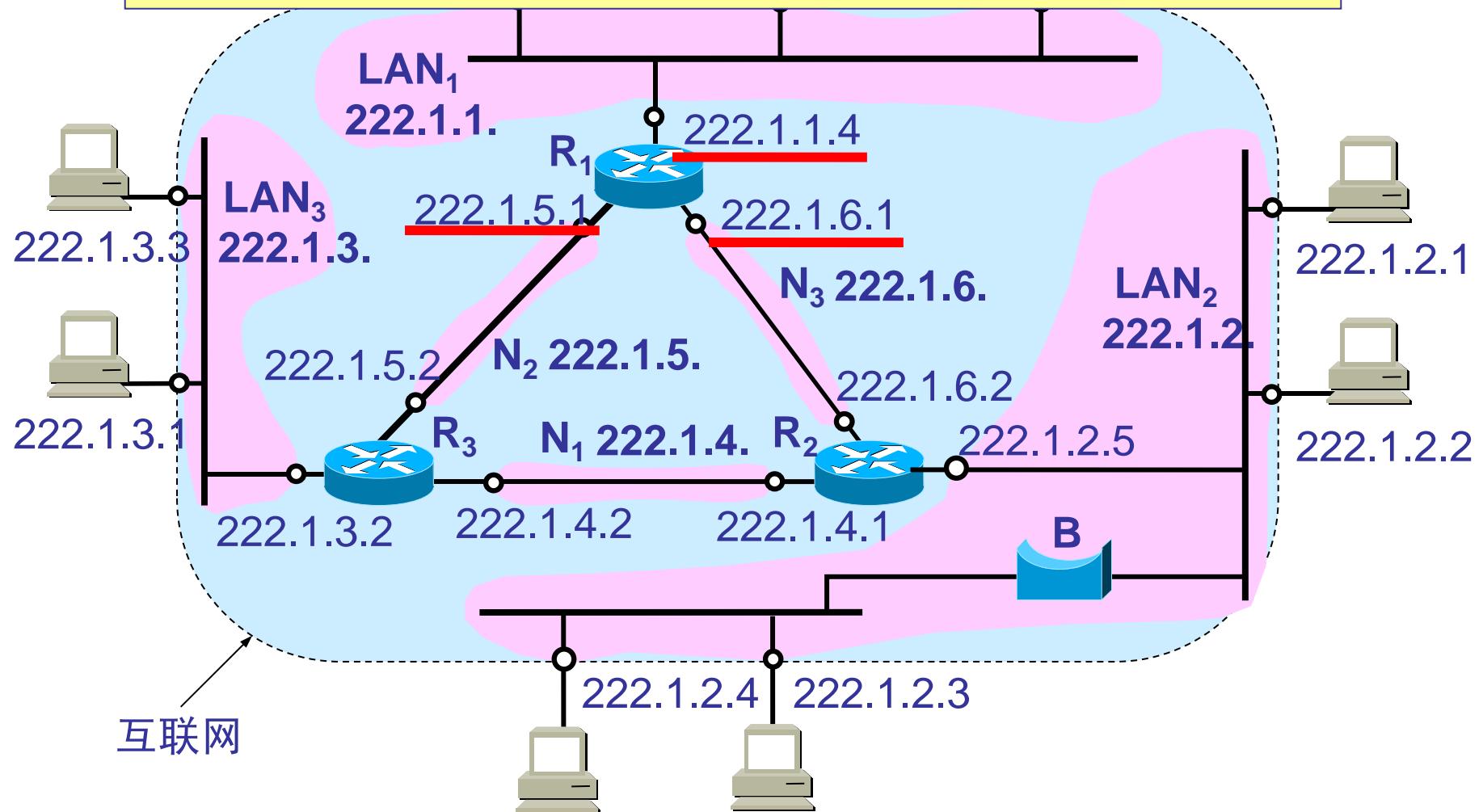
互联网中的 IP 地址



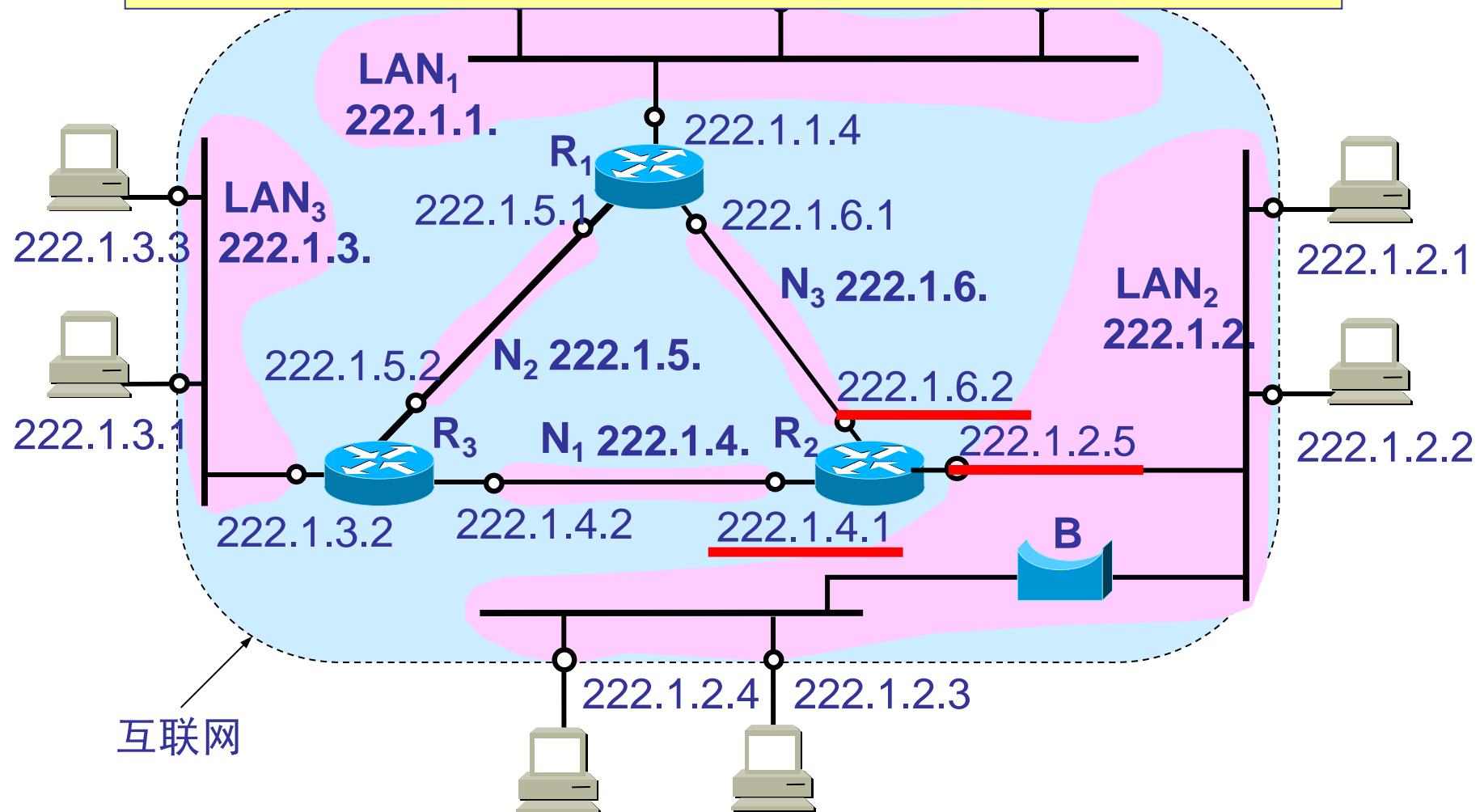
在同一个局域网上的主机或路由器的
IP 地址中的网络号必须是一样的。
图中的网络号就是 IP 地址中的 net-id



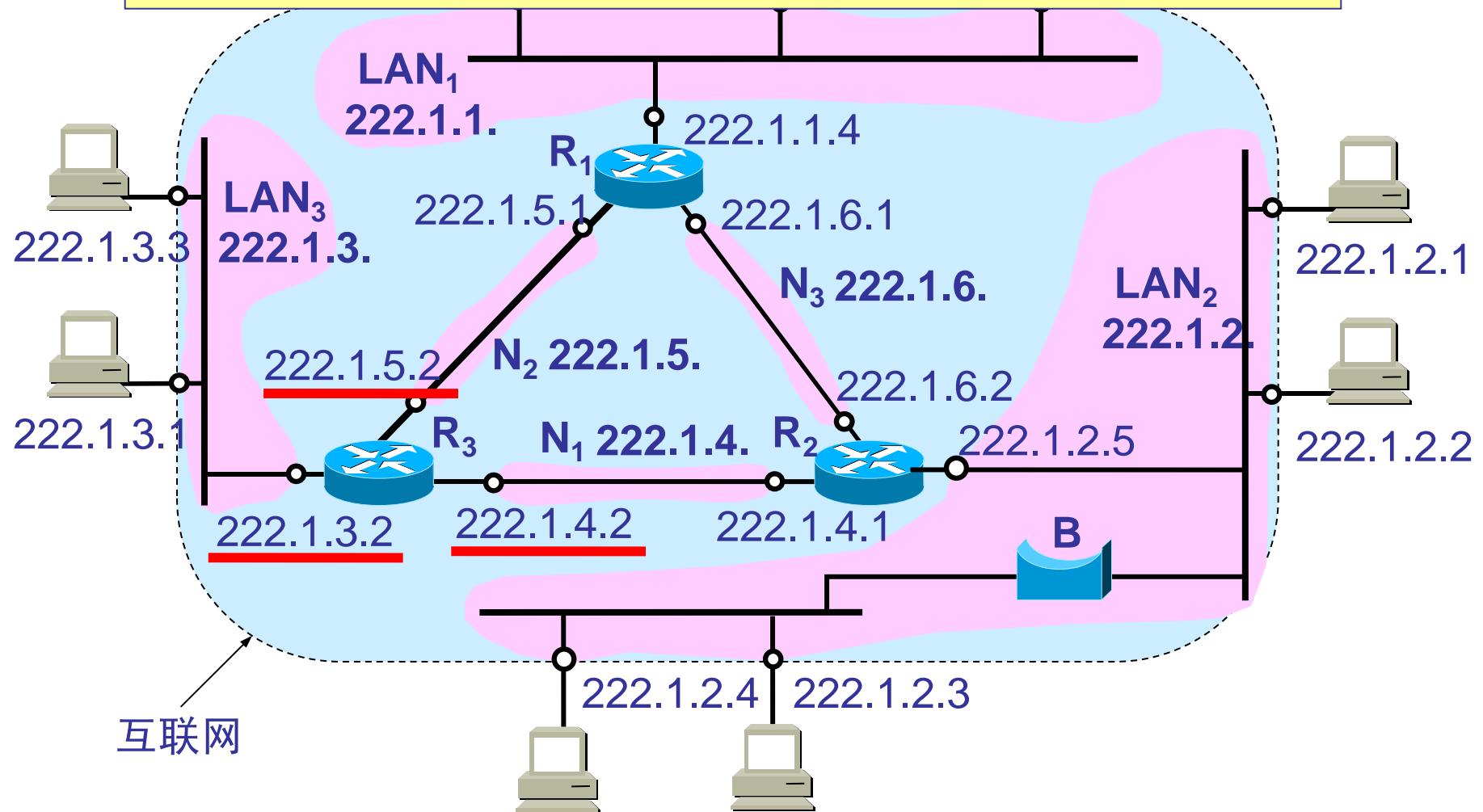
路由器总是具有两个或两个以上的 IP 地址。
路由器的每一个接口都有一个
不同网络号的 IP 地址。



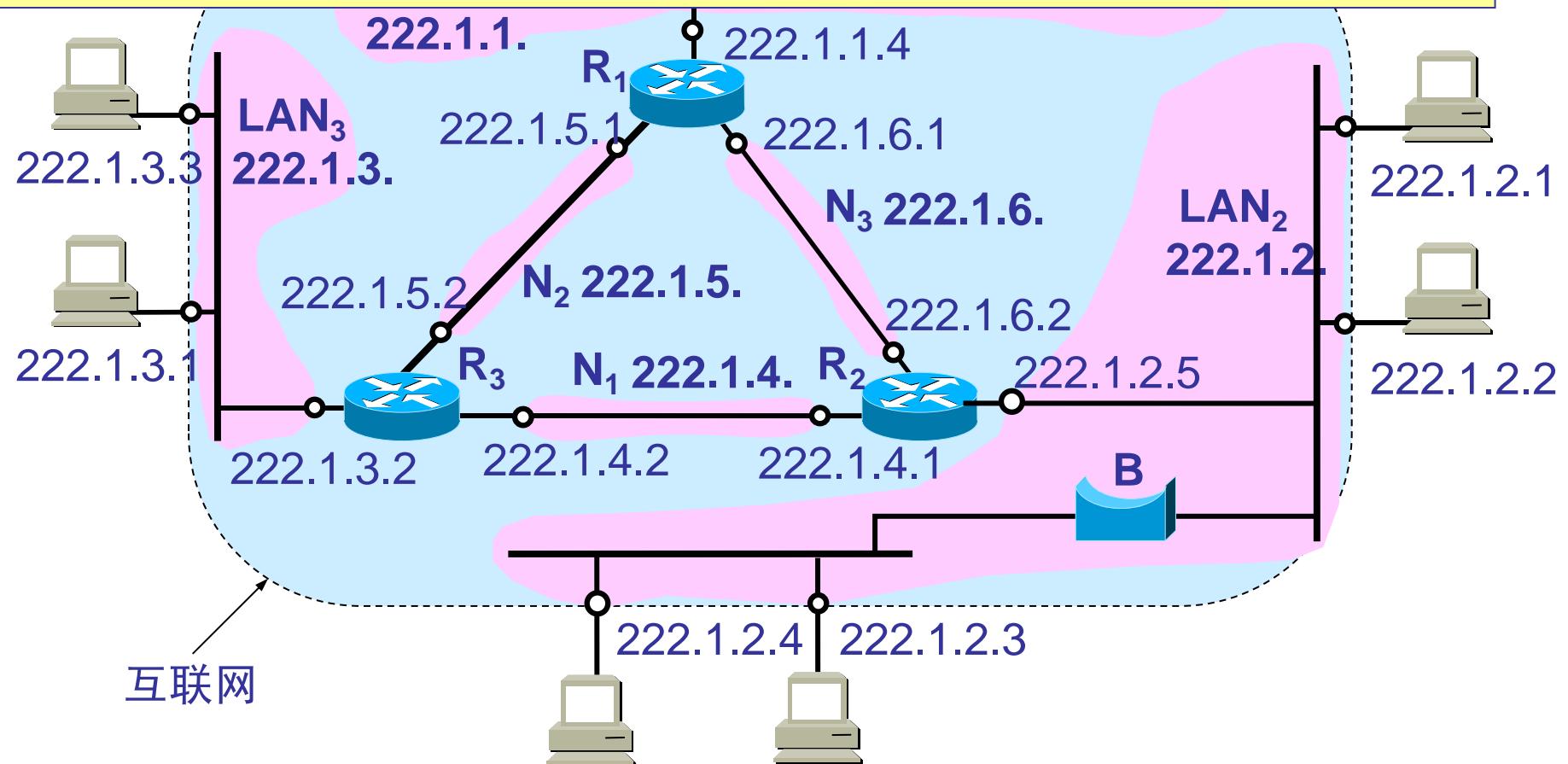
路由器总是具有两个或两个以上的 IP 地址。
路由器的每一个接口都有一个
不同网络号的 IP 地址。

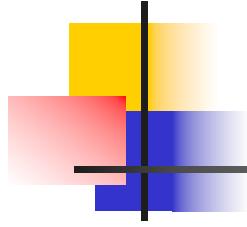


路由器总是具有两个或两个以上的 IP 地址。
路由器的每一个接口都有一个
不同网络号的 IP 地址。



两个路由器直接相连的接口处，可指明也可不指明 IP 地址。如指明 IP 地址，则这一段连线就构成了一种只包含一段线路的特殊“网络”。现在常不指明 IP 地址。

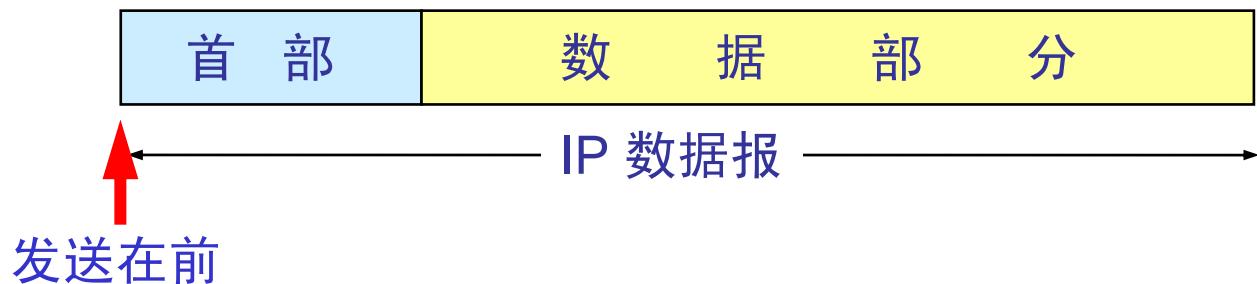




4.2.5 IP 数据报的格式

- 一个 IP 数据报由首部和数据两部分组成。
- 首部的前一部分是固定长度，共 20 字节，是所有 IP 数据报必须具有的。
- 在首部的固定部分的后面是一些可选字段，其长度是可变的。







1. IP 数据报头部的固定部分中的各字段



版本——占 4 位，指 IP 协议的版本
目前的 IP 协议版本号为 4 (即 IPv4)



首部长度——占 4 位，可表示的最大数值
 是 15 个单位(一个单位为 4 字节)
 因此 IP 的首部长度的最大值是 60 字节。



区分服务——占 8 位，用来获得更好的服务
 在旧标准中叫做**服务类型**，但实际上一直未被使用过。
 1998 年这个字段改名为**区分服务**。
 只有在使用区分服务（DiffServ）时，这个字段才起作用。
 在一般的情况下都不使用这个字段



总长度——占 16 位，指首部和数据之和的长度，
 单位为字节，因此数据报的最大长度为 65535 字节。
 总长度必须不超过最大传送单元 MTU。



标识(identification) 占 16 位，
它是一个计数器，用来产生数据报的标识。



标志(flag) 占 3 位，目前只有前两位有意义。

标志字段的最低位是 **MF** (More Fragment)。

$MF = 1$ 表示后面“还有分片”。 $MF = 0$ 表示最后一个分片。

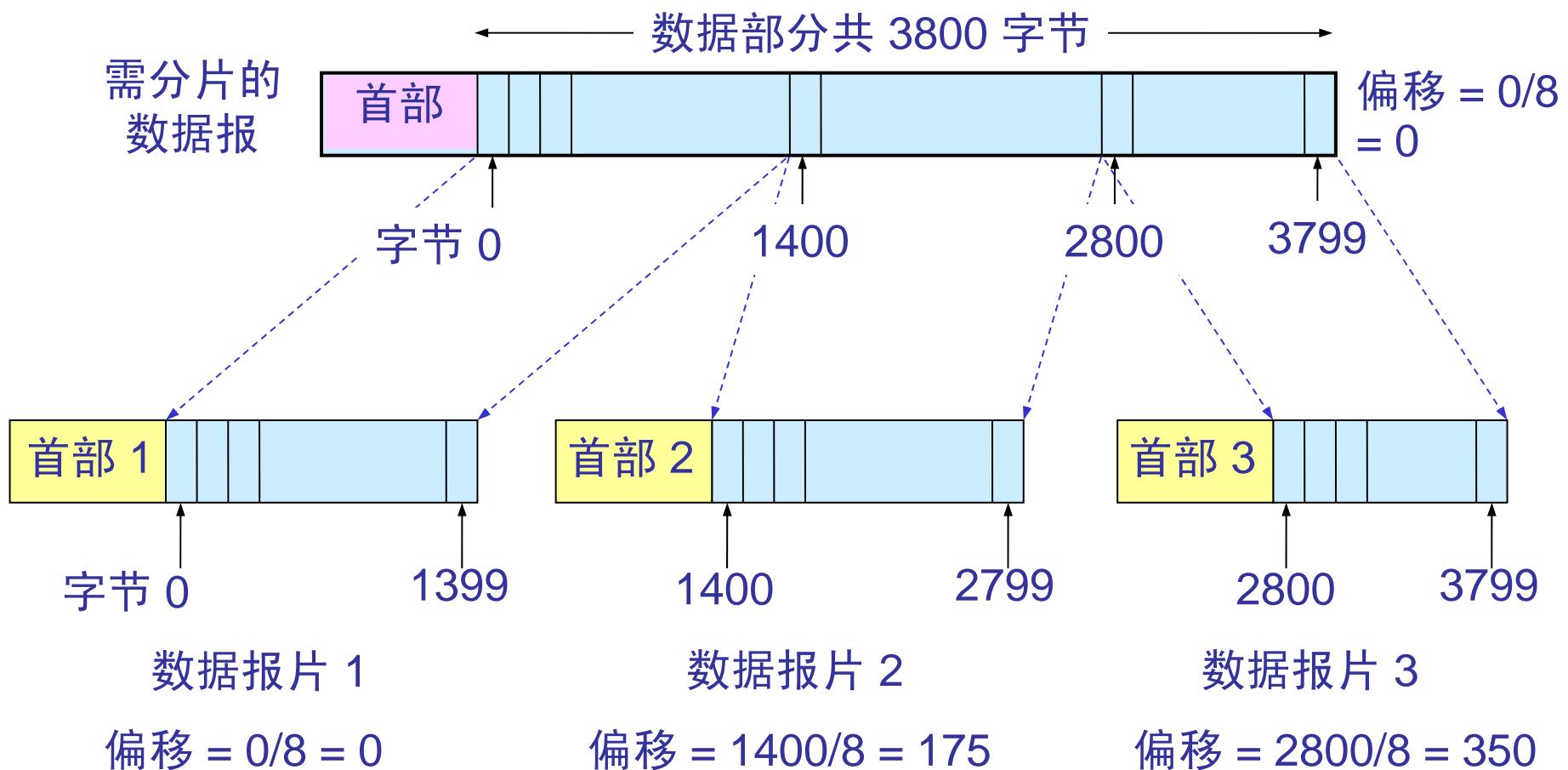
标志字段中间的一位是 **DF** (Don't Fragment)。

只有当 $DF = 0$ 时才允许分片。



片偏移(12位)指出：较长的分组在分片后
某片在原分组中的相对位置。
片偏移以 8 个字节为偏移单位。

【例4-1】IP 数据报分片

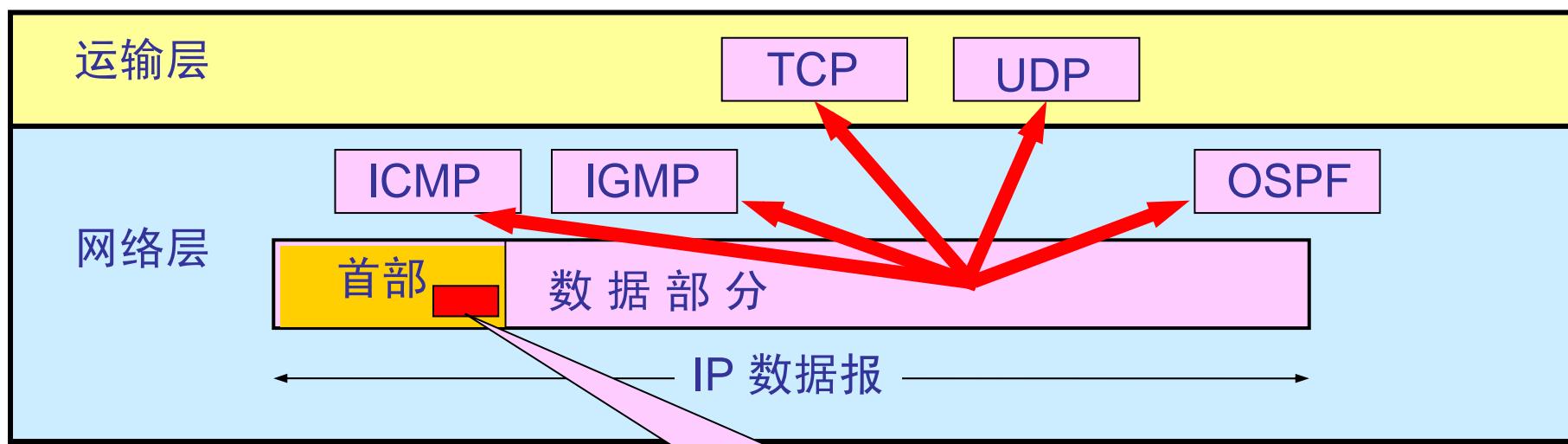




生存时间(8位)记为 TTL (Time To Live)
数据报在网络中可通过的路由器数的最大值。



协议(8位)字段指出此数据报携带的数据使用何种协议以便目的主机的IP层将数据部分上交给哪个处理过程

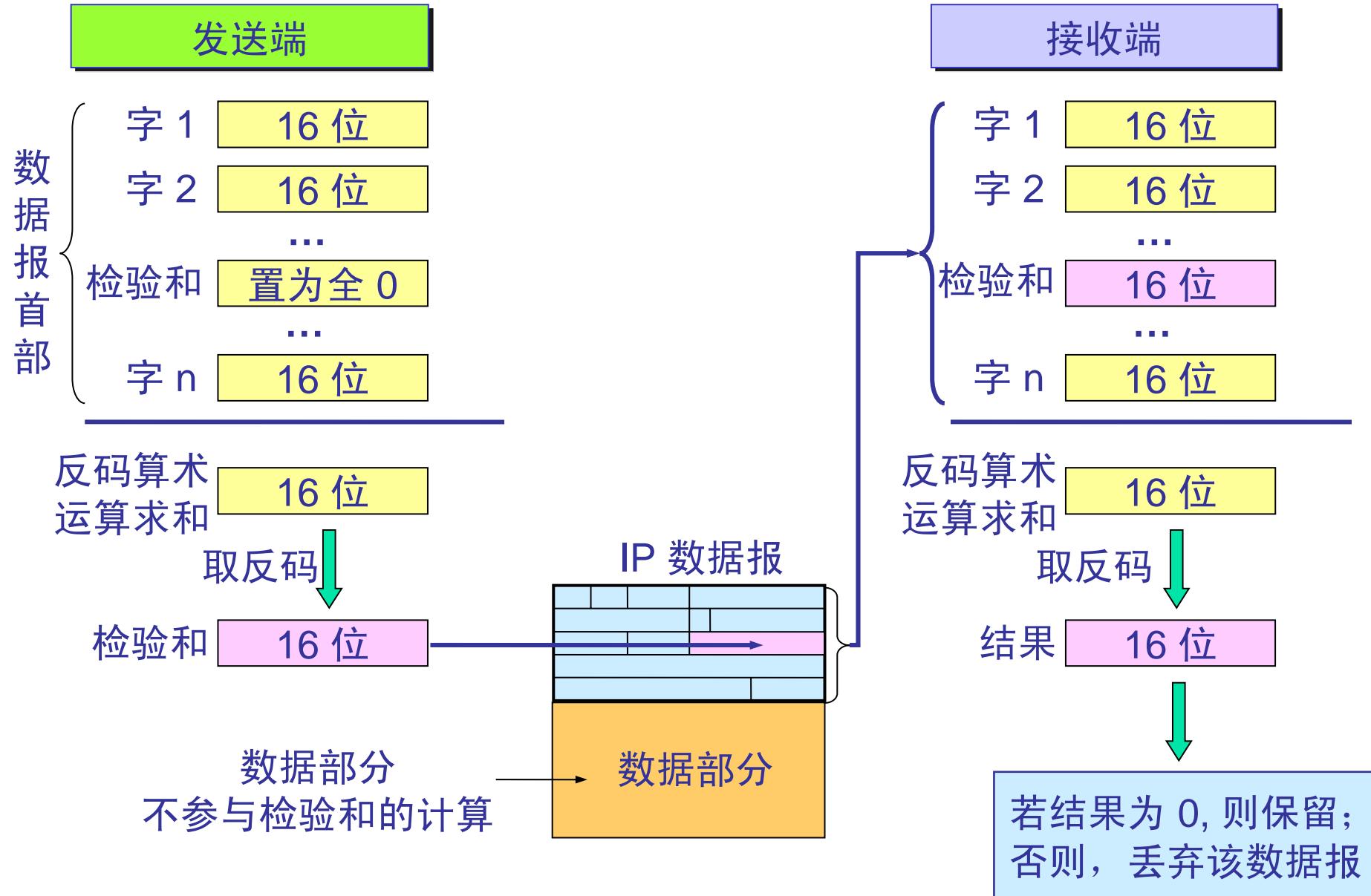


协议字段指出应将数据部分交给哪一个进程



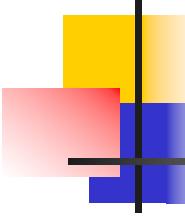
首部检验和(16位)字段只检验数据报的首部
不检验数据部分。

这里不采用 CRC 检验码而采用简单的计算方法。



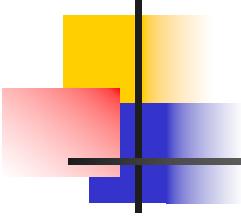


源地址和目的地址都各占 4 字节



2. IP 数据报头部的可变部分

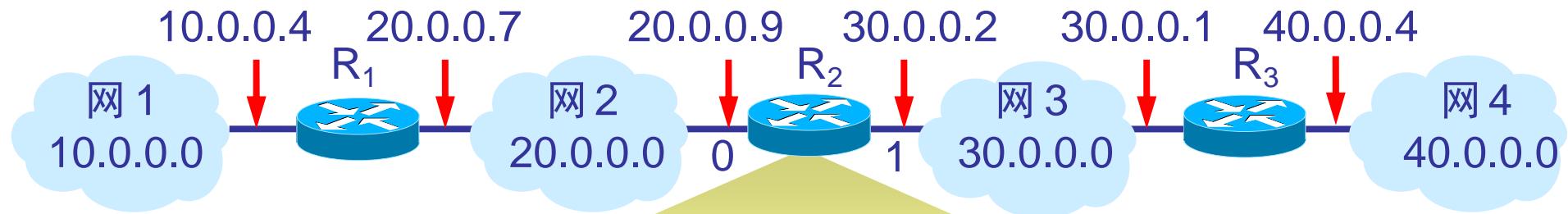
- IP 首部的可变部分就是一个选项字段，用来支持排错、测量以及安全等措施，内容很丰富。
- 选项字段的长度可变，从 1 个字节到 40 个字节不等，取决于所选择的项目。
- 增加头部的可变部分是为了增加 IP 数据报的功能，但这同时也使得 IP 数据报的首部长度成为可变的。这就增加了每一个路由器处理数据报的开销。
- 实际上这些选项很少被使用。



4.2.6 IP 层转发分组的流程

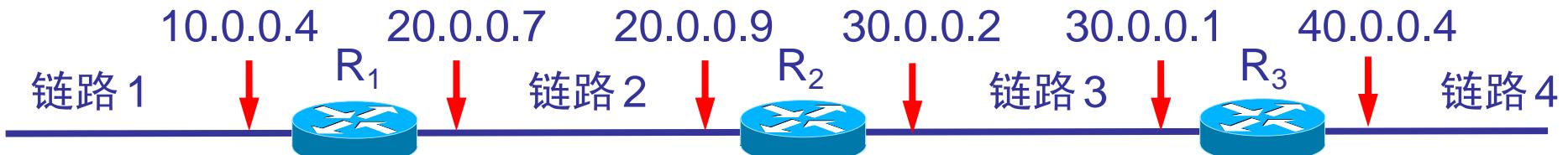
- 有四个 A 类网络通过三个路由器连接在一起。每一个网络上都可能有成千上万个主机。
- 可以想像，若按目的主机号来制作路由表，则所得出的路由表就会过于庞大。
- 但若按主机所在的**网络地址**来制作路由表，那么每一个路由器中的路由表就只包含 4 个项目。这样就可使路由表大大简化。

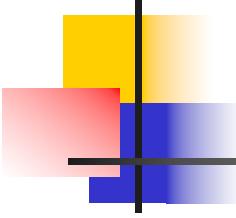
在路由表中，对每一条路由，最主要的是
(目的网络地址，下一跳地址)



路由器 R₂ 的路由表

目的主机所在的网络	下一跳地址
20.0.0.0	直接交付, 接口 0
30.0.0.0	直接交付, 接口 1
10.0.0.0	20.0.0.7
40.0.0.0	30.0.0.1

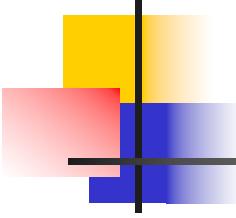




查找路由表

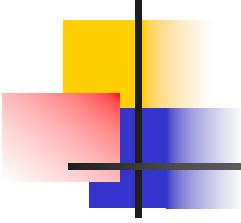
根据目的网络地址就能确定下一跳路由器，
这样做的结果是：

- IP 数据报最终一定可以找到目的主机所在目的网络上的路由器（可能要通过多次的间接交付）。
- 只有到达最后一个路由器时，才试图向目的主机进行直接交付。



特定主机路由

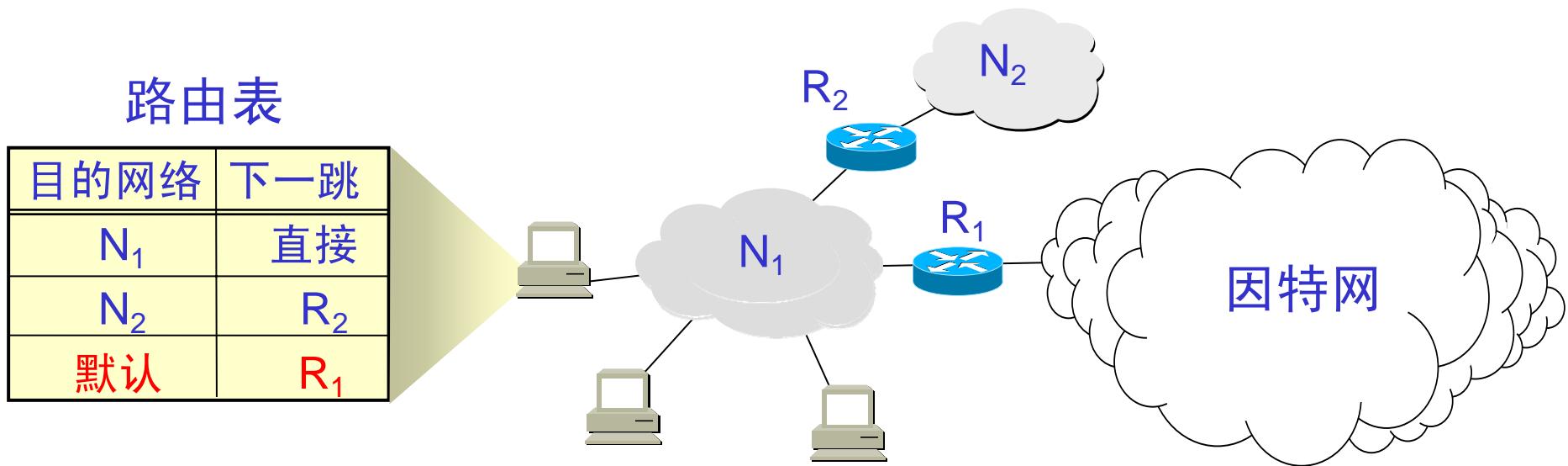
- 这种路由是为特定的目的主机指明一个路由。
- 采用特定主机路由可使网络管理人员能更方便地控制网络和测试网络，同时也可在需要考虑某种安全问题时采用这种特定主机路由。

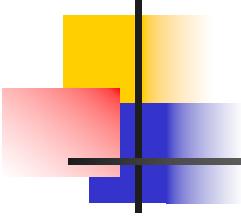


默认路由(default route)

- 路由器还可采用**默认路由**以减少路由表所占用的空间和搜索路由表所用的时间。
- 这种转发方式在一个网络只有很少的对外连接时是很有用的。
- 默认路由在主机发送 IP 数据报时往往更能显示出它的好处。
- 如果一个主机连接在一个小网络上，而这个网络只用一个路由器和因特网连接，那么在这种情况下使用默认路由是非常合适的。

只要目的网络不是 N_1 和 N_2 ，
就一律选择默认路由，
把数据报先间接交付路由器 R_1 ，
让 R_1 再转发给下一个路由器。



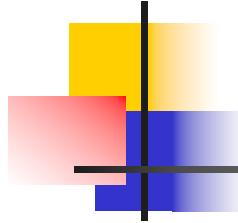


必须强调指出

- IP 数据报的头部中没有地方可以用来指明“下一跳路由器的 IP 地址”。
- 当路由器收到待转发的数据报，不是将下一跳路由器的 IP 地址填入 IP 数据报，而是送交下层的网络接口软件。
- 网络接口软件使用 ARP 负责将下一跳路由器的 IP 地址转换成硬件地址，并将此硬件地址放在链路层的 MAC 帧的头部，然后根据这个硬件地址找到下一跳路由器。

分组转发算法

- (1) 从数据报的头部提取目的主机的 IP 地址 D , 得出目的网络地址为 N 。
- (2) 若网络 N 与此路由器直接相连, 则把数据报直接交付目的主机 D ; 否则是间接交付, 执行(3)。
- (3) 若路由表中有目的地址为 D 的特定主机路由, 则把数据报传送给路由表中所指明的下一跳路由器; 否则, 执行(4)。
- (4) 若路由表中有到达网络 N 的路由, 则把数据报传送给路由表指明的下一跳路由器; 否则, 执行(5)。
- (5) 若路由表中有一个默认路由, 则把数据报传送给路由表中所指明的默认路由器; 否则, 执行(6)。
- (6) 报告转发分组出错。



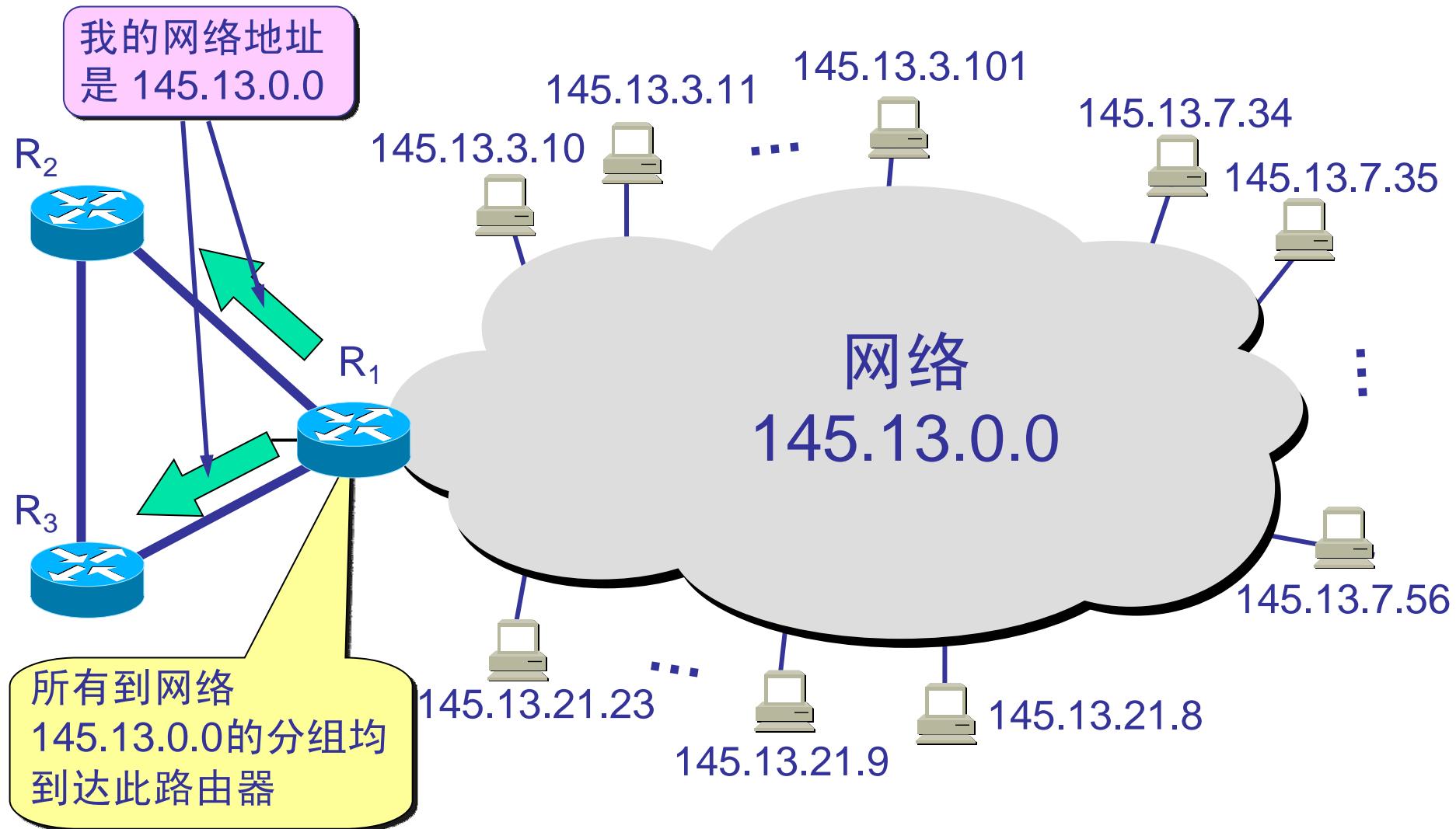
4.3 划分子网和构造超网

4.3.1 划分子网

1. 从两级 IP 地址到三级 IP 地址

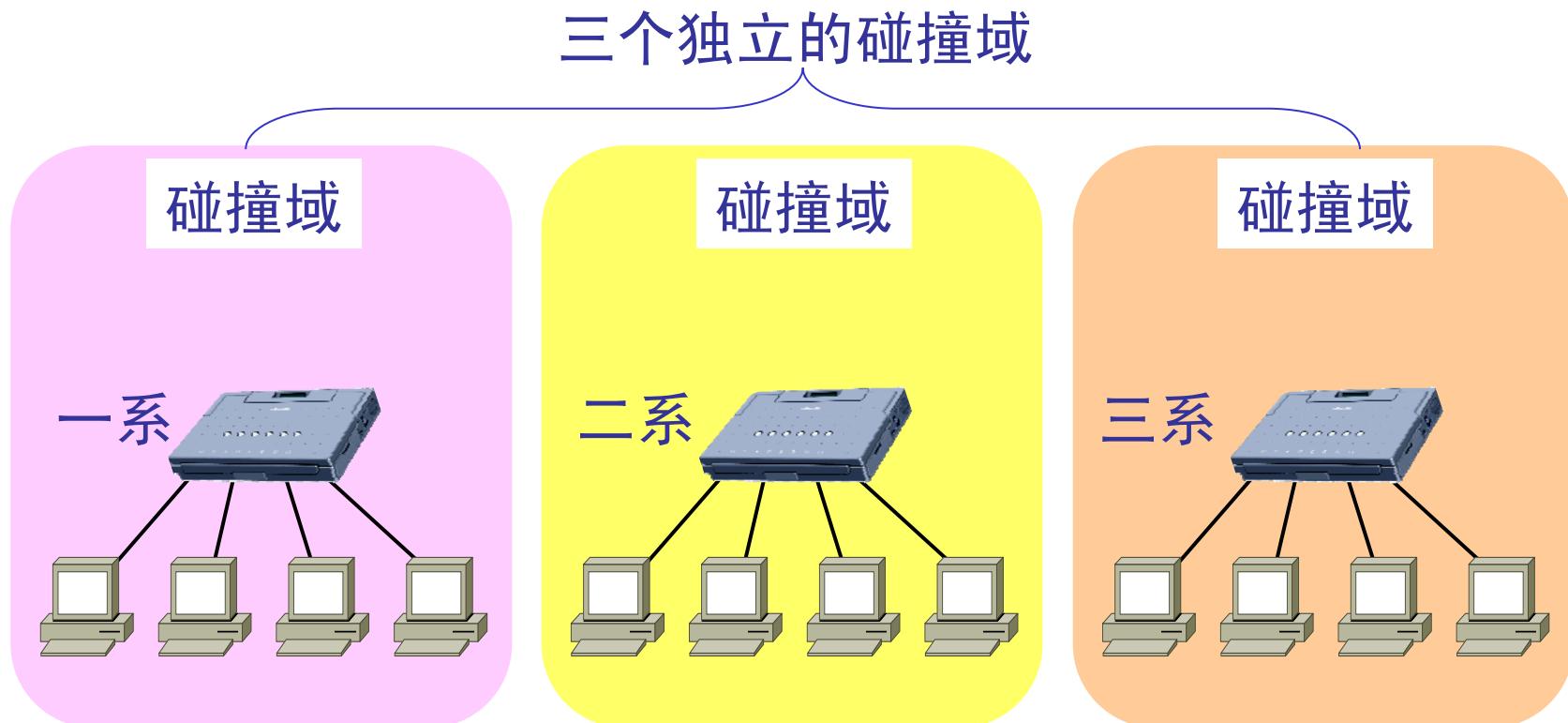
- 在 ARPANET 的早期，IP 地址的设计确实不够合理。
 - IP 地址空间的利用率有时很低。
 - 给每一个物理网络分配一个网络号会使路由表变得太大因而使网络性能变坏。
 - 两级的 IP 地址不够灵活。

一个未划分子网的 B 类网络 145.13.0.0

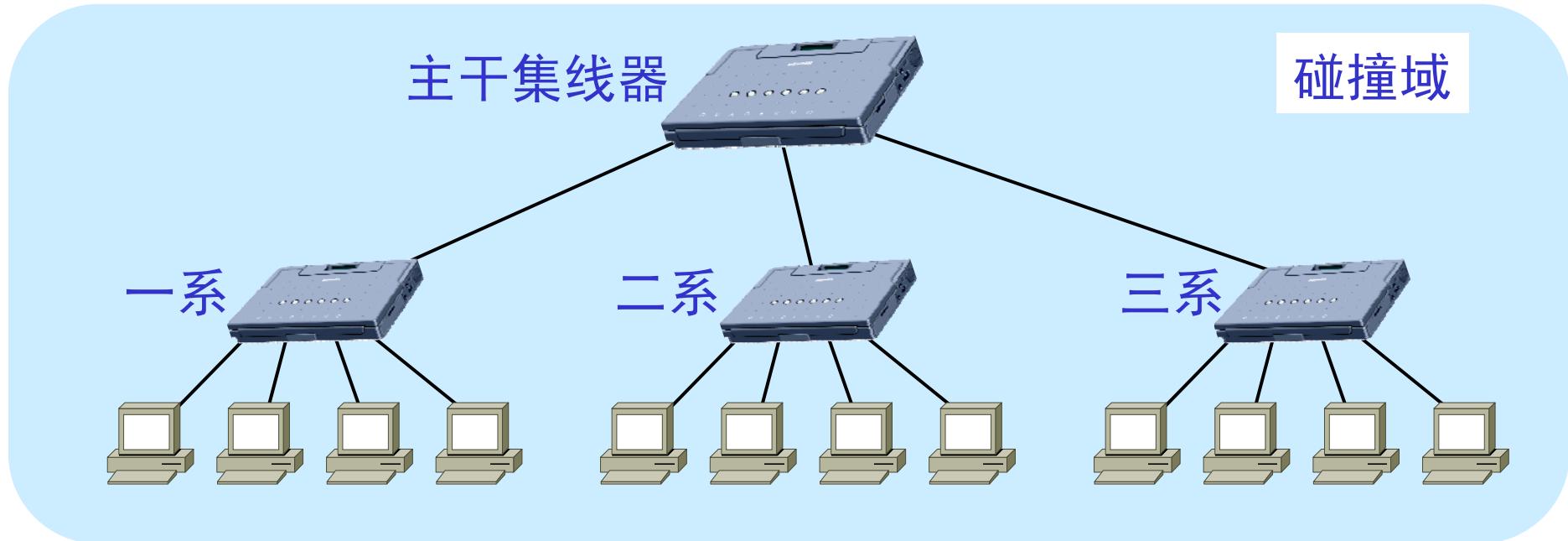


B类网络遇到的扩展问题

- 某大学有三个系，各自有一个局域网

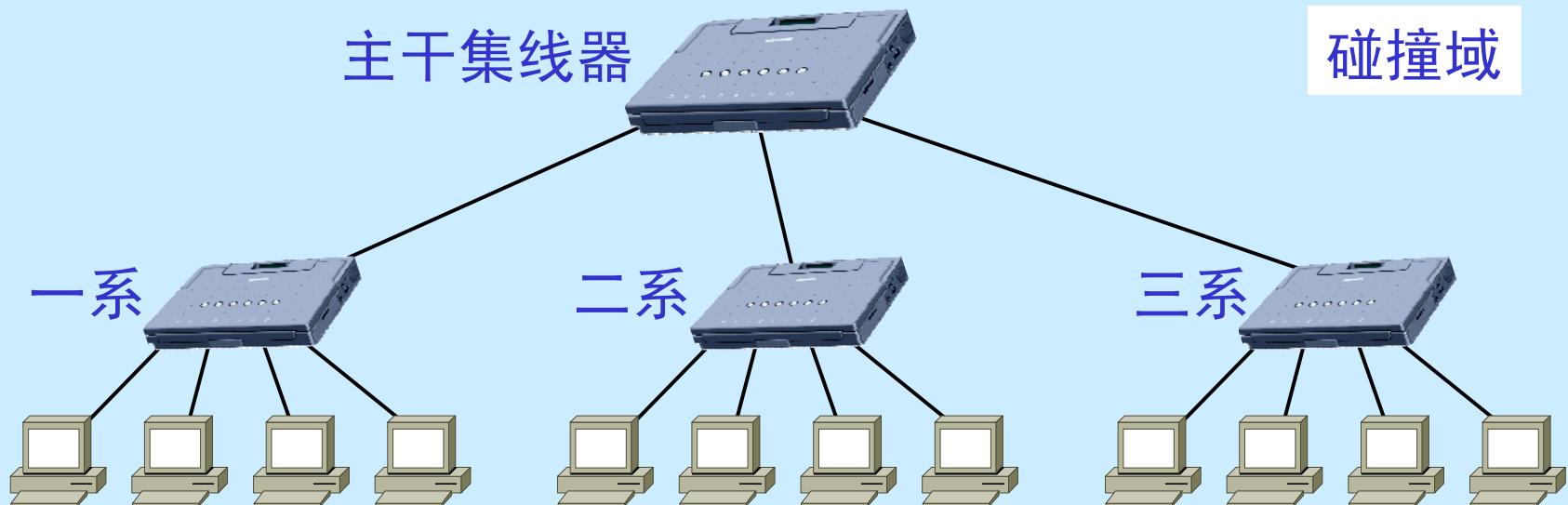


B类网络遇到的扩展问题



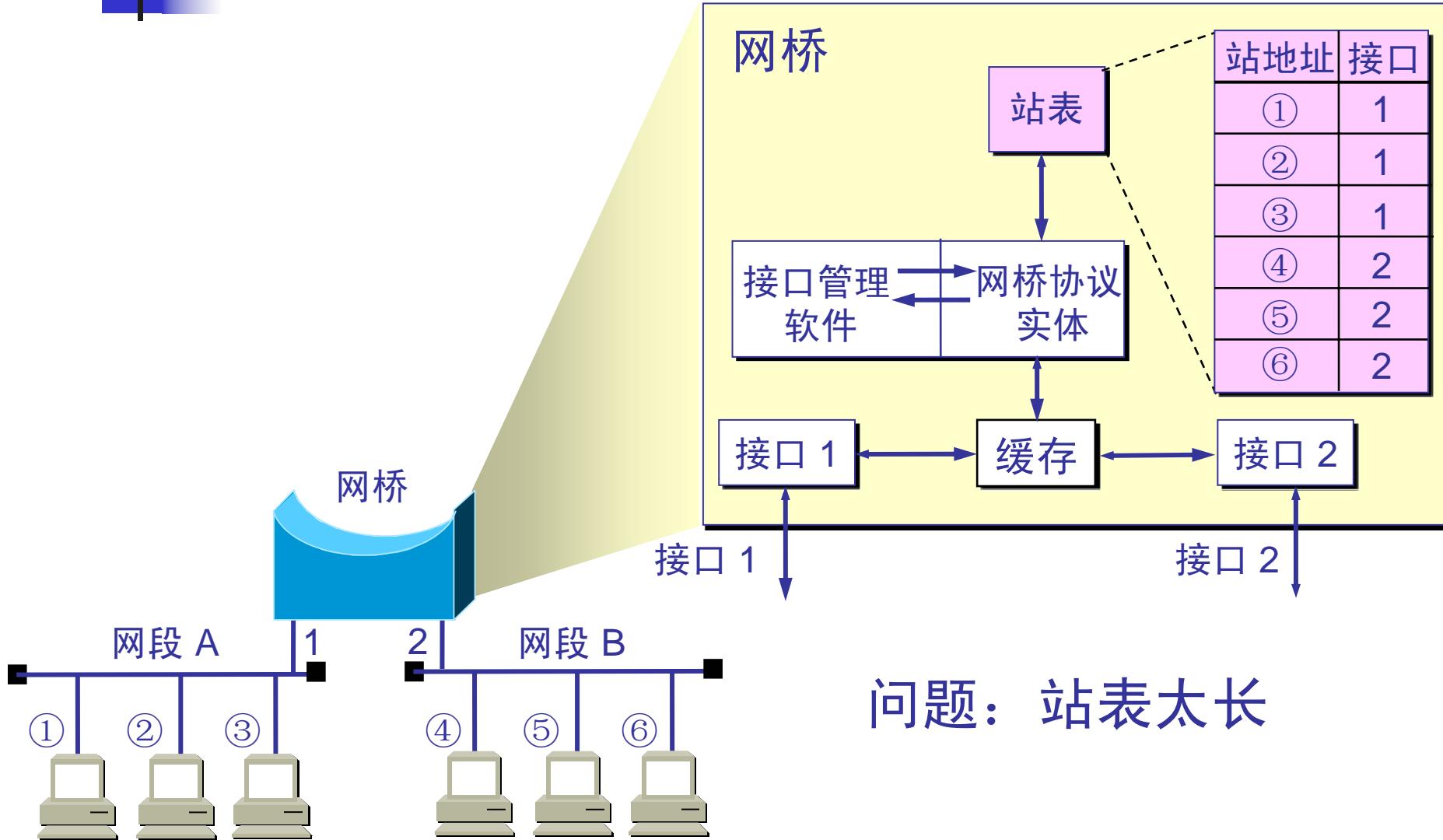
问题：碰撞域太大， 网络性能大大降低。

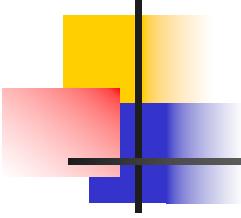
B类网络遇到的扩展问题



问题：碰撞域太大， 网络性能大大降低。

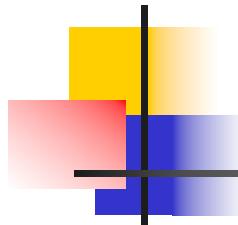
B类网络遇到的扩展问题





三级的 IP 地址

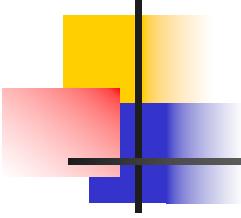
- 从 1985 年起在 IP 地址中又增加了一个“子网号字段”，使两级的 IP 地址变成为**三级的 IP 地址**。
- 这种做法叫作**划分子网**(subnetting)。划分子网已成为因特网的正式标准协议。



划分子网的基本思路

- 划分子网纯属一个单位内部的事情。单位对外仍然表现为没有划分子网的网络。
- 从主机号借用若干个位作为子网号 subnet-id，而主机号 host-id 也就相应减少了若干个位。

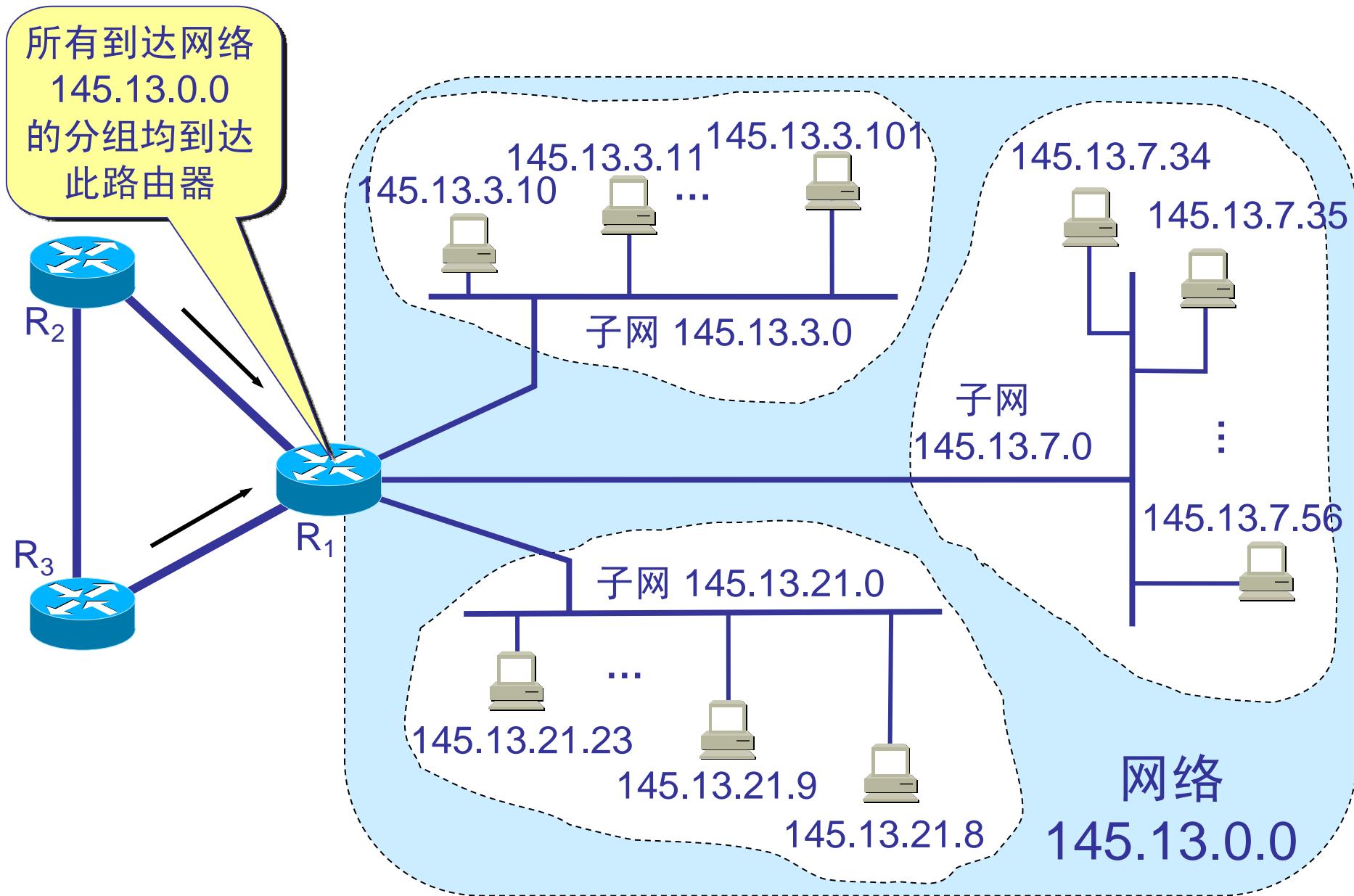
IP地址 ::= {<网络号>, <子网号>, <主机号>} (4-2)

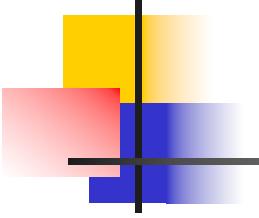


划分子网的基本思路（续）

- 凡是从其他网络发送给本单位某个主机的 IP 数据报，仍然是根据 IP 数据报的**目的网络号** net-id，先找到连接在**本单位网络上的路由器**。
- 然后**此路由器**在收到 IP 数据报后，再按**目的网络号 net-id 和子网号 subnet-id** 找到**目的子网**。
- 最后就将 IP 数据报直接交付**目的主机**。

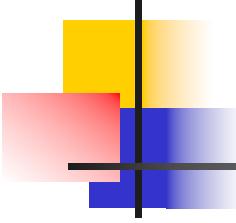
划分为三个子网后对外仍是一个网络





划分子网后变成了三级结构

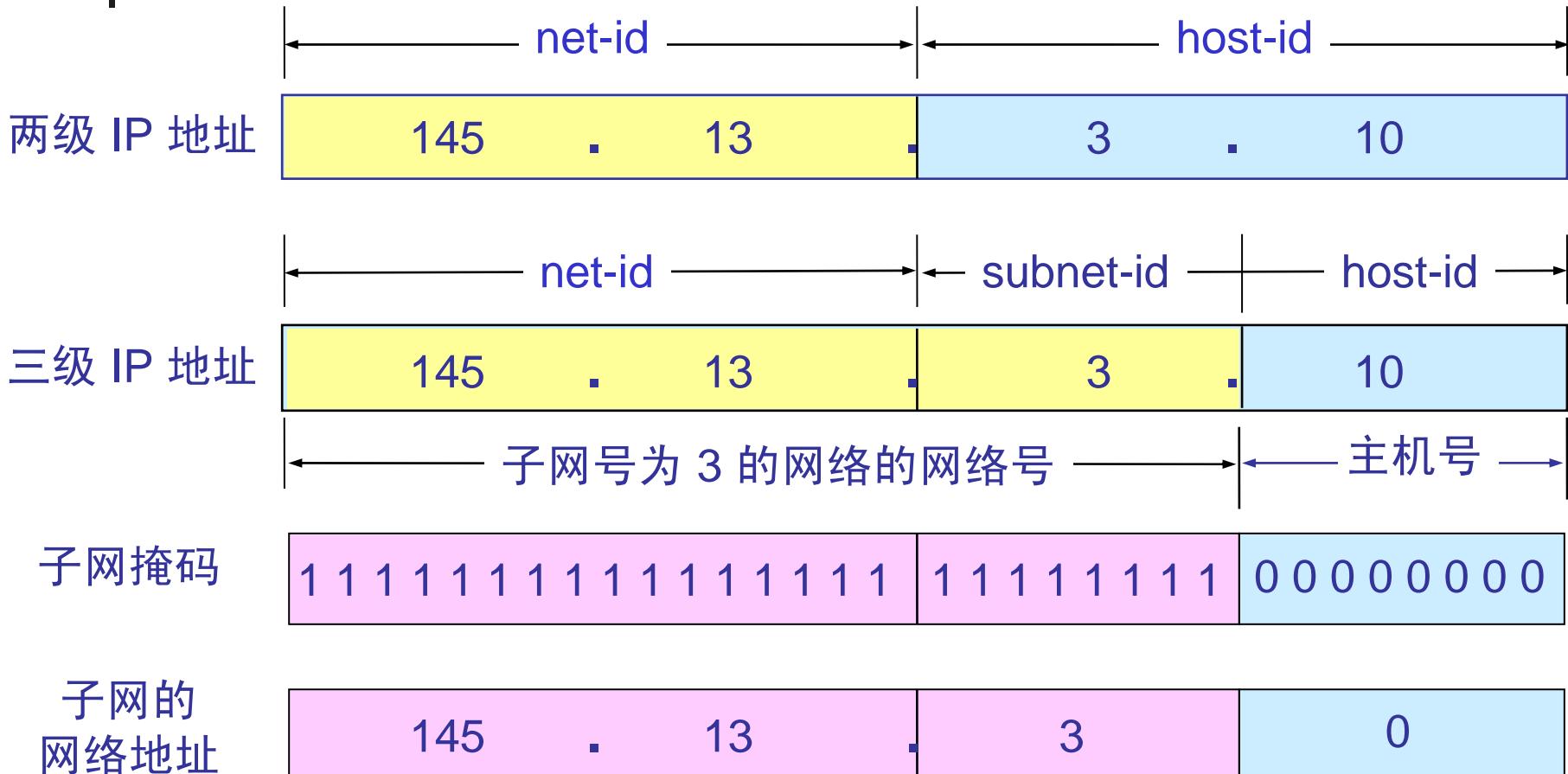
- 当没有划分子网时，IP 地址是两级结构。
- 划分子网后 IP 地址就变成了**三级结构**。
- 划分子网只是把 IP 地址的主机号 host-id 这部分进行再划分，而不改变 IP 地址原来的网络号 net-id。

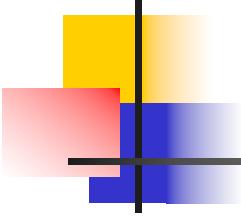


2. 子网掩码

- 从一个 IP 数据报的头部并无法判断源主机或目的主机所连接的网络是否进行了子网划分。
- 使用**子网掩码**(subnet mask)可以找出 IP 地址中的子网部分。

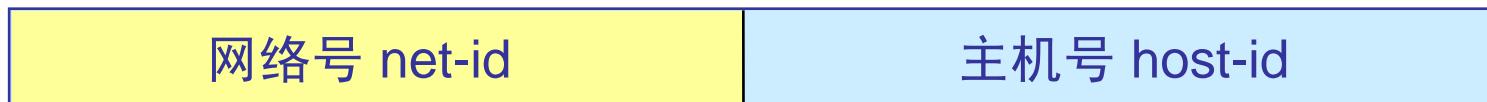
IP 地址的各字段和子网掩码



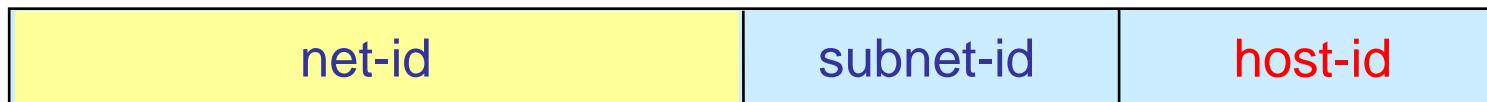


(IP 地址) AND (子网掩码) = 网络地址

两级 IP 地址



三级 IP 地址

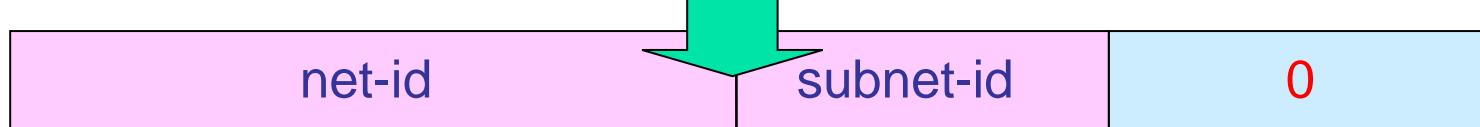


逐位进行 AND 运算

子网掩码

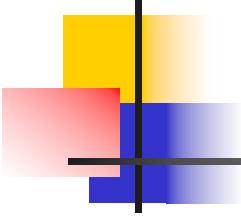


子网的
网络地址



默认子网掩码

A类地址	网络地址	net-id	host-id 为全 0
	默认子网掩码 255.0.0.0		
B类地址	网络地址	net-id	host-id 为全 0
	默认子网掩码 255.255.0.0		
C类地址	网络地址	net-id	host-id 为全 0
	默认子网掩码 255.255.255.0		



子网掩码是一个重要属性

- 子网掩码是一个网络或一个子网的重要属性。
- 路由器在和相邻路由器交换路由信息时，必须把自己所在网络（或子网）的子网掩码告诉相邻路由器。
- 路由器的路由表中的每一个项目，除了要给出目的网络地址外，还必须同时给出该网络的子网掩码。
- 若一个路由器连接在两个子网上就拥有两个网络地址和两个子网掩码。

【例4-2】已知 IP 地址是 141.14.72.24，子网掩码是 255.255.192.0。试求网络地址。

(a) 点分十进制表示的 IP 地址

141	.	14	.	72	.	24
-----	---	----	---	----	---	----

(b) IP 地址的第 3 字节是二进制

141	.	14	.01001000.	24
-----	---	----	------------	----

(c) 子网掩码是 255.255.192.0

11111111	11111111	11000000	00000000
----------	----------	----------	----------

(d) IP 地址与子网掩码逐位相与

141	.	14	.01000000.	0
-----	---	----	------------	---

(e) 网络地址（点分十进制表示）

141	.	14	.	64	.	0
-----	---	----	---	----	---	---

【例4-3】 在上例中，若子网掩码改为 255.255.224.0。试求网络地址，讨论所得结果。

(a) 点分十进制表示的 IP 地址

141	.	14	.	72	.	24
-----	---	----	---	----	---	----

(b) IP 地址的第 3 字节是二进制

141	.	14	.01001000.	24
-----	---	----	------------	----

(c) 子网掩码是 255.255.224.0

11111111	11111111	11100000	00000000
----------	----------	----------	----------

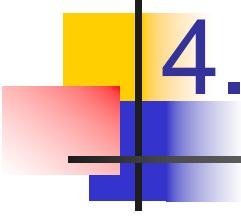
(d) IP 地址与子网掩码逐位相与

141	.	14	.01000000.	0
-----	---	----	------------	---

(e) 网络地址（点分十进制表示）

141	.	14	.	64	.	0
-----	---	----	---	----	---	---

不同的子网掩码得出**相同的**网络地址。
但不同的掩码的效果是不同的。



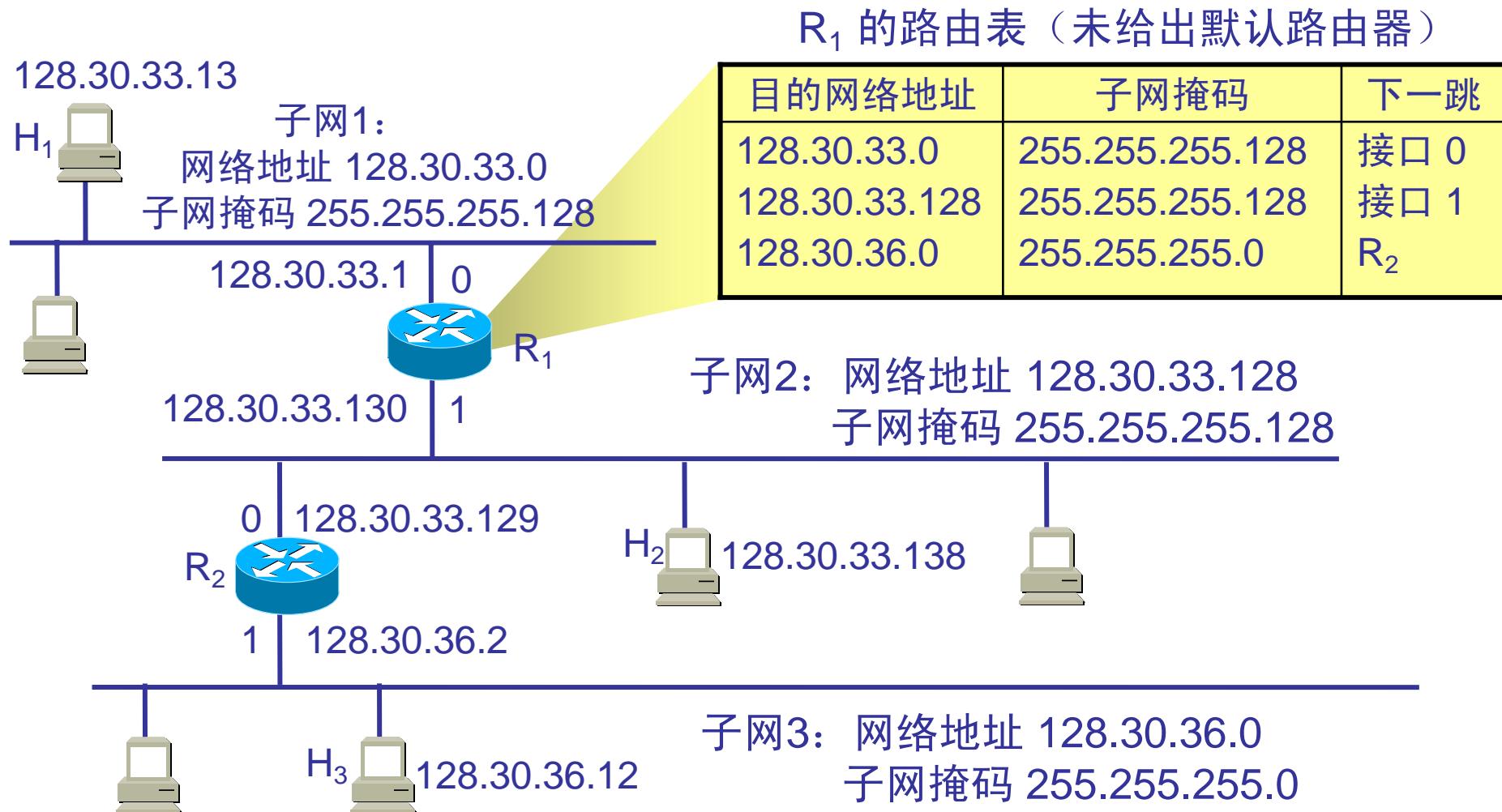
4.3.2 使用子网掩码的分组转发过程

- 在不划分子网的两级 IP 地址下，从 IP 地址得出网络地址是个很简单的事。
- 但在划分子网的情况下，从 IP 地址却不能唯一地得出网络地址来，这是因为网络地址取决于那个网络所采用的子网掩码，但数据报的首部并没有提供子网掩码的信息。
- 因此分组转发的算法也必须做相应的改动。

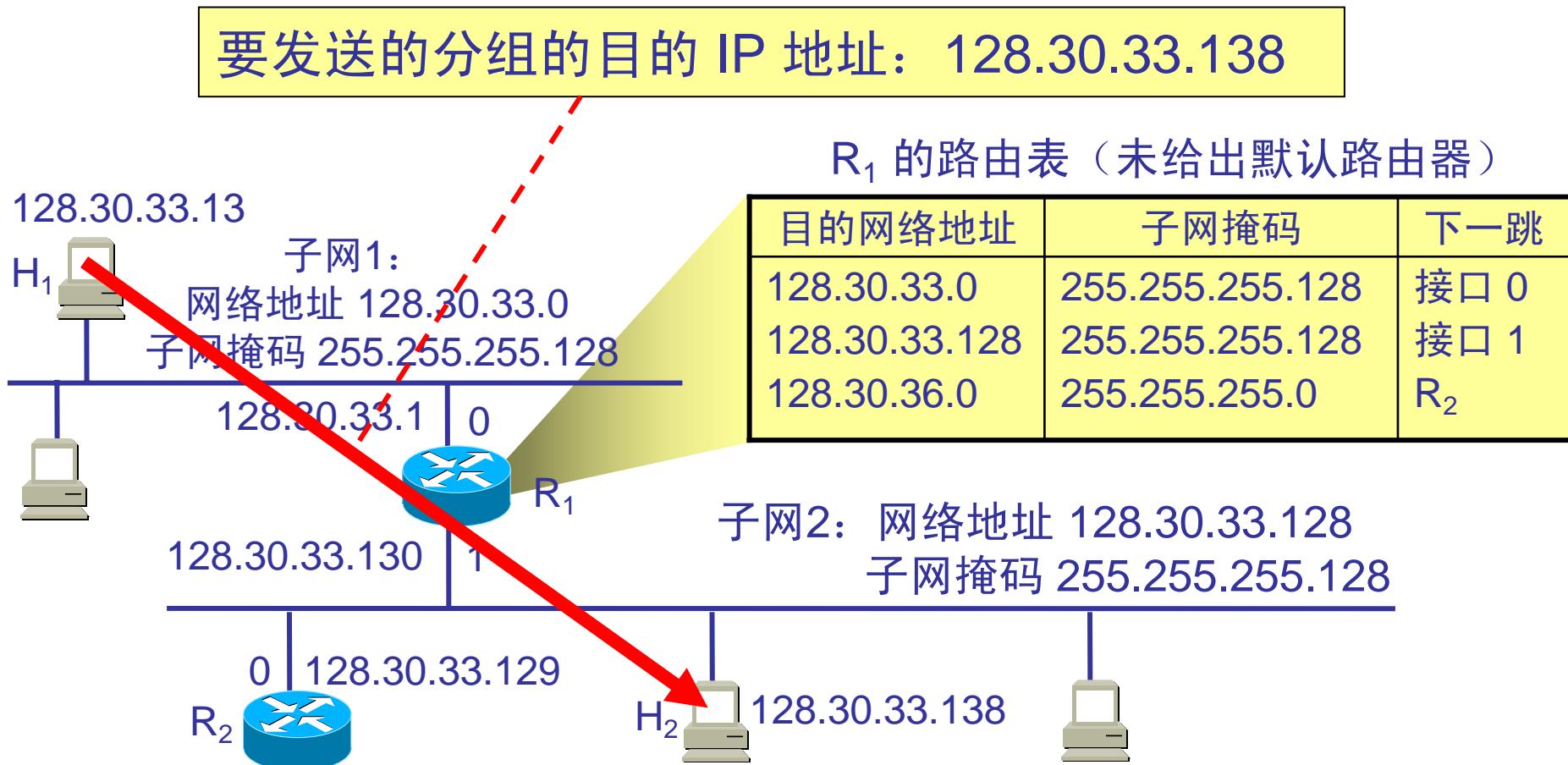
在划分子网的情况下路由器转发分组的算法

- (1) 从收到的分组的首部提取目的 IP 地址 D 。
- (2) 先用各网络的子网掩码和 D 逐位相“与”，看是否和相应的网络地址匹配。若匹配，则将分组直接交付。否则就是间接交付，执行(3)。
- (3) 若路由表中有目的地址为 D 的特定主机路由，则将分组传送给指明的下一跳路由器；否则，执行(4)。
- (4) 对路由表中的每一行的子网掩码和 D 逐位相“与”，若其结果与该行的目的网络地址匹配，则将分组传送给该行指明的下一跳路由器；否则，执行(5)。
- (5) 若路由表中有一个默认路由，则将分组传送给路由表中所指明的默认路由器；否则，执行(6)。
- (6) 报告转发分组出错。

【例4-4】已知互联网和路由器 R_1 中的路由表。主机 H_1 向 H_2 发送分组。试讨论 R_1 收到 H_1 向 H_2 发送的分组后查找路由表的过程。



主机 H_1 要发送分组给 H_2



主机 H_1 首先将
本子网的子网掩码 255.255.255.128
与分组的 IP 地址 128.30.33.138 逐比特相“与”(AND 操作)

255.255.255.128 AND 128.30.33.138 的计算

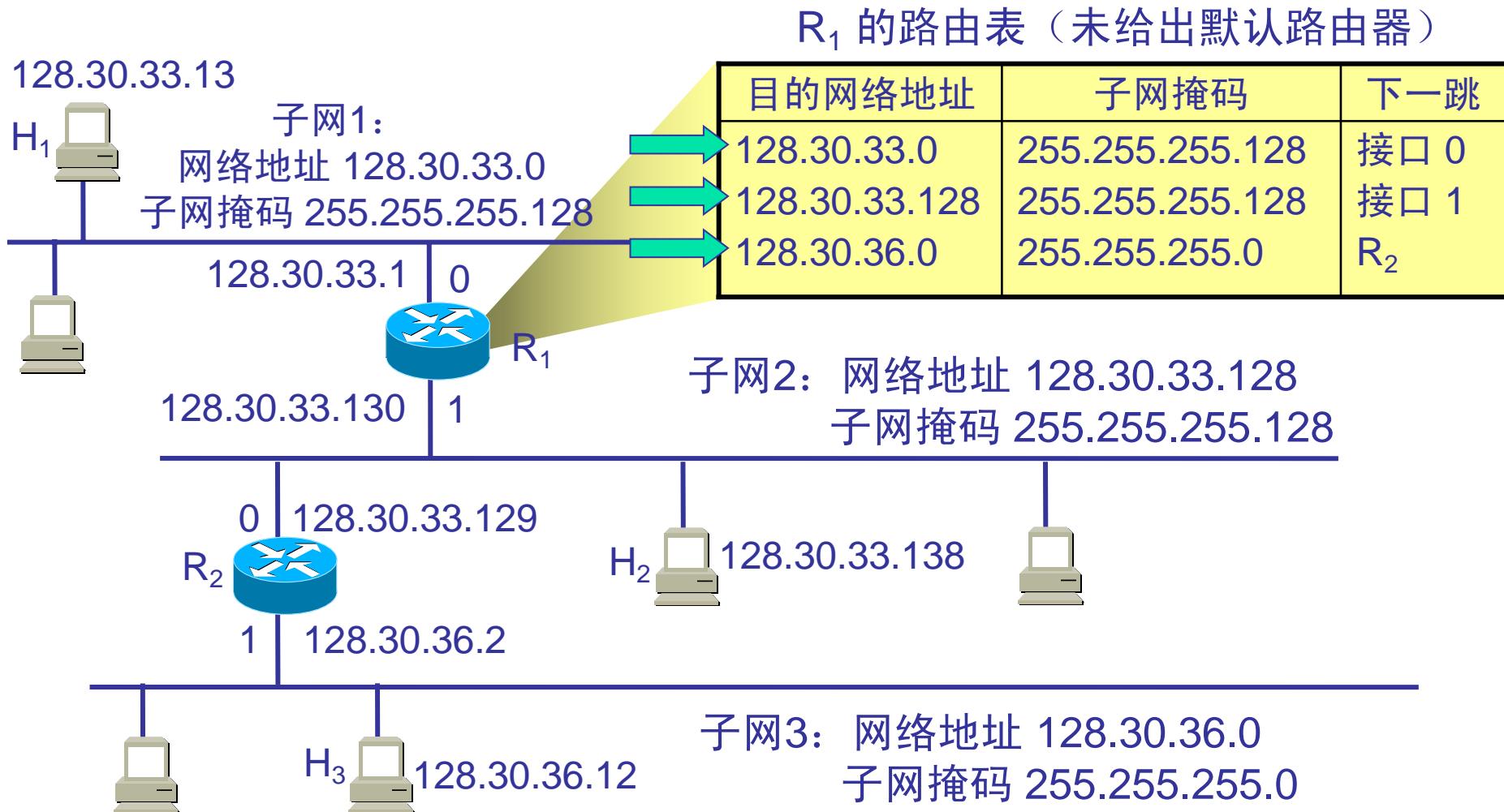
255 就是二进制的全 1，因此 255 AND xyz = xyz，
这里只需计算最后的 128 AND 138 即可。

$$\begin{array}{r} 128 \rightarrow 10000000 \\ 138 \rightarrow 10001010 \\ \hline \end{array}$$

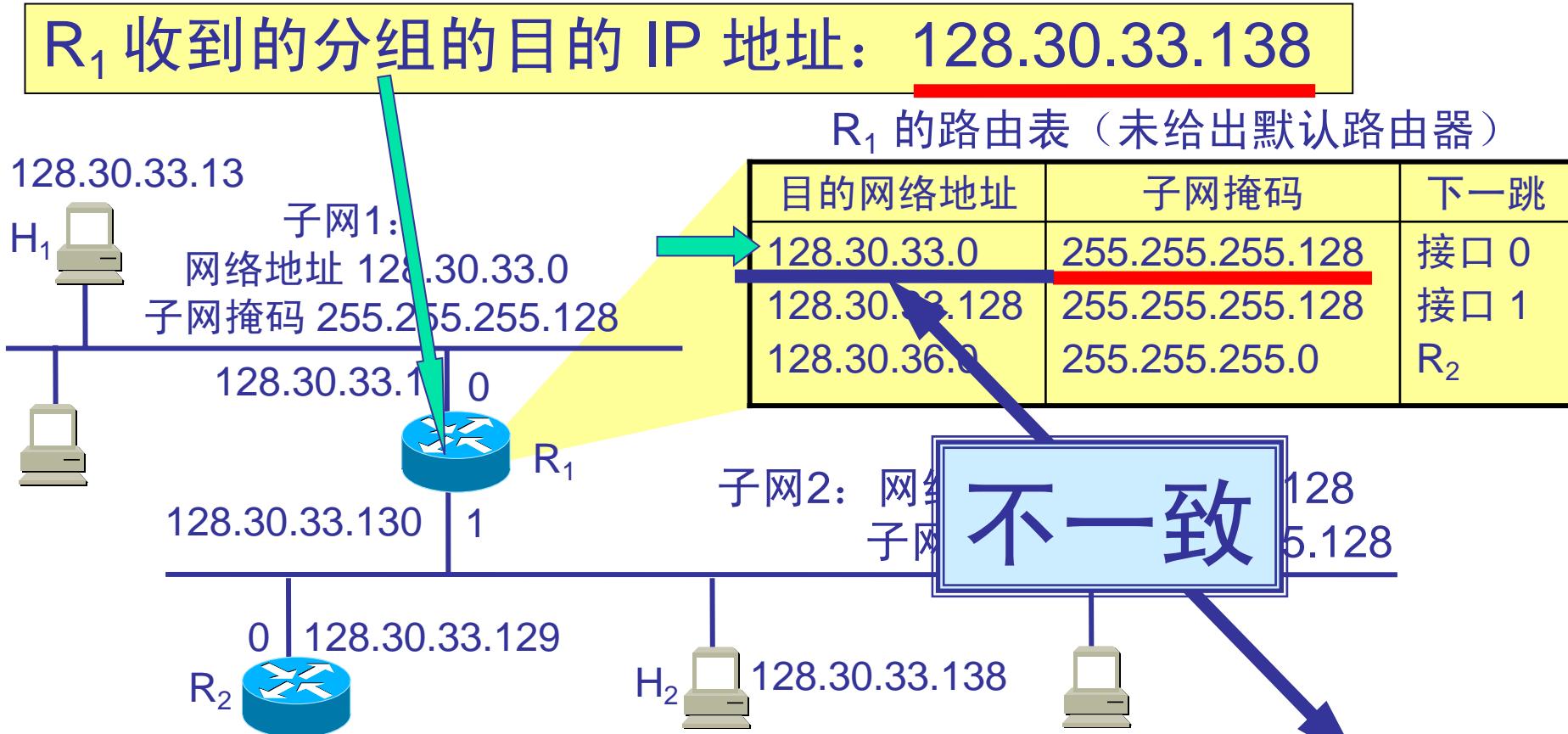
逐比特 AND 操作后: 10000000 → 128

$$\begin{array}{r} 255.255.255.128 \\ 128. 30. 33.138 \\ \hline 128. 30. 33.128 \neq H_1 \text{ 的网络地址} \end{array}$$

因此 H_1 必须把分组传送到路由器 R_1
然后逐项查找路由表



路由器 R_1 收到分组后就用路由表中第 1 个项目的子网掩码和 128.30.33.138 逐比特 AND 操作



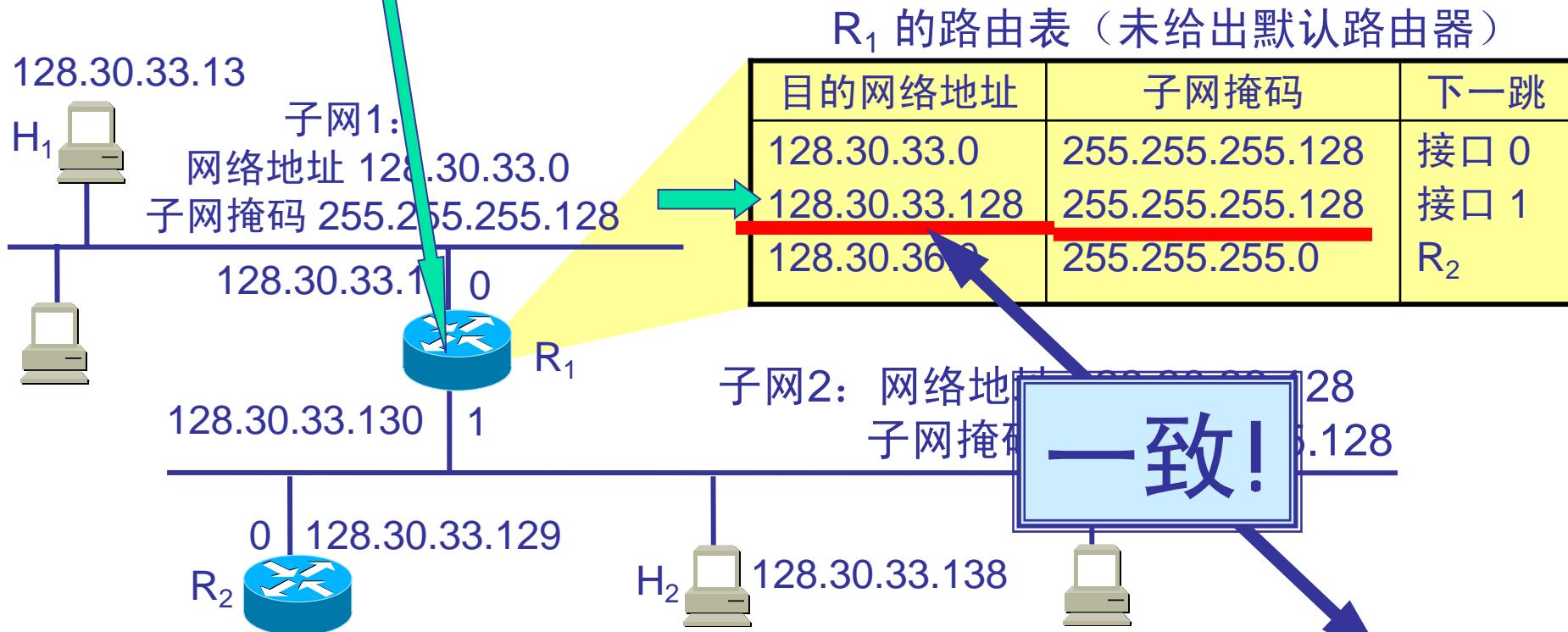
$$255.255.255.128 \text{ AND } 128.30.33.138 = 128.30.33.128$$

不匹配!

(因为 128.30.33.128 与路由表中的 128.30.33.0 不一致)

路由器 R_1 再用路由表中第 2 个项目的子网掩码和 128.30.33.138 逐比特 AND 操作

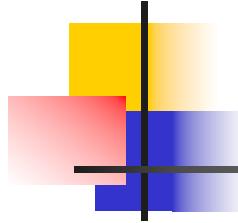
R_1 收到的分组的目的 IP 地址: 128.30.33.138



$$255.255.255.128 \text{ AND } 128.30.33.138 = 128.30.33.128$$

匹配!

这表明子网 2 就是收到的分组所要寻找的目的网络

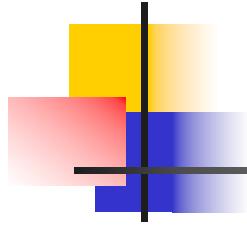


4.3.3 无分类编址 CIDR

1. 网络前缀

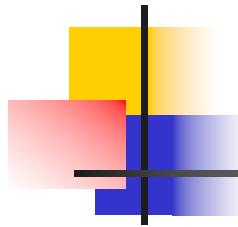
划分子网在一定程度上缓解了因特网在发展中遇到的困难。然而在 1992 年因特网仍然面临三个必须尽早解决的问题，这就是：

- B 类地址在 1992 年已分配了近一半，眼看就要在 1994 年 3 月全部分配完毕！
- 因特网主干网上的路由表中的项目数急剧增长（从几千个增长到几万个）。
- 整个 IPv4 的地址空间最终将全部耗尽。



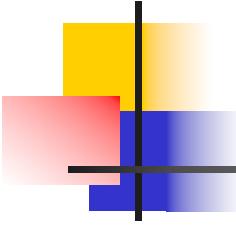
IP 编址问题的演进

- 1987 年，RFC 1009 就指明了在一个划分子网的网络中可同时使用几个不同的子网掩码。使用 **变长子网掩码 VLSM** (Variable Length Subnet Mask) 可进一步提高 IP 地址资源的利用率。
- 在 VLSM 的基础上又进一步研究出无分类编址方法，它的正式名字是 **无分类域间路由选择 CIDR** (Classless Inter-Domain Routing)。



CIDR 最主要的特点

- CIDR 消除了传统的 A 类、B 类和 C 类地址以及划分子网的概念，因而可以更加有效地分配 IPv4 的地址空间。
- CIDR 使用各种长度的“**网络前缀**”(network-prefix)来代替分类地址中的网络号和子网号。
- IP 地址从三级编址（使用子网掩码）又回到了两级编址。

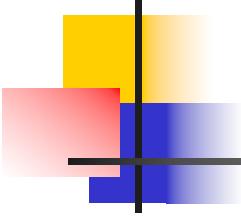


无分类的两级编址

- 无分类的两级编址的记法是：

IP地址 ::= {<网络前缀>, <主机号>} (4-3)

- CIDR 还使用“斜线记法”(slash notation)，它又称为**CIDR记法**，即在 IP 地址面加上一个斜线“/”，然后写上网络前缀所占的位数（这个数值对应于三级编址中子网掩码中 1 的个数）。
- CIDR 把网络前缀都相同的连续的 IP 地址组成“**CIDR 地址块**”。



CIDR 地址块

- 128.14.32.0/20 表示的地址块共有 2^{12} 个地址（因为斜线后面的 20 是网络前缀的位数，所以这个地址的主机号是 12 位）。
- 这个地址块的起始地址是 128.14.32.0。
- 在不需要指出地址块的起始地址时，也可将这样的地址块简称为“/20 地址块”。
- 128.14.32.0/20 地址块的最小地址：128.14.32.0
- 128.14.32.0/20 地址块的最大地址：128.14.47.255
- 全 0 和全 1 的主机号地址一般不使用。

128.14.32.0/20 表示的地址 (2^{12} 个地址)

最小地址

10000000 00001110 00100000 00000000

10000000 00001110 00100000 00000001

10000000 00001110 00100000 00000010

10000000 00001110 00100000 00000011

10000000 00001110 00100000 00000100

10000000 00001110 00100000 00000101

...

...

10000000 00001110 00101111 11111011

10000000 00001110 00101111 11111100

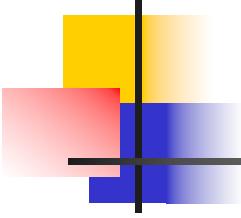
10000000 00001110 00101111 11111101

10000000 00001110 00101111 11111110

10000000 00001110 00101111 11111111

最大地址

所有地址
的 20 位
前缀都是
一样的



路由聚合(route aggregation)

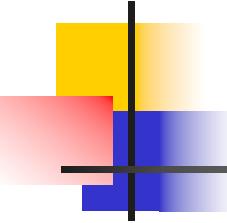
- 一个 CIDR 地址块可以表示很多地址，这种地址的聚合常称为**路由聚合**，它使得路由表中的一个项目可以表示很多个（例如上千个）原来传统分类地址的路由。
- 路由聚合也称为**构成超网(supernetting)**。
- CIDR 虽然不使用子网了，但仍然使用“**掩码**”这一名词（但不叫子网掩码）。
- 对于 /20 地址块，它的掩码是 20 个连续的 1。斜线记法中的数字就是掩码中 1 的个数。

CIDR 记法的其他形式

- 10.0.0.0/10 可简写为 10/10，也就是把点分十进制中低位连续的 0 省略。
- 10.0.0.0/10 隐含地指出 IP 地址 10.0.0.0 的**掩码**是 255.192.0.0。此掩码可表示为

11111111 11000000 00000000 00000000
 └──┘ └──┘ └──┘ └──┘
 255 192 0 0

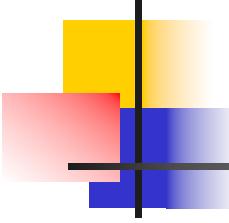
掩码中有 10 个连续的 1



CIDR 记法的其他形式

- 10.0.0.0/10 可简写为 10/10，也就是将点分十进制中低位连续的 0 省略。
- 10.0.0.0/10 相当于指出 IP 地址 10.0.0.0 的掩码是 255.192.0.0，即

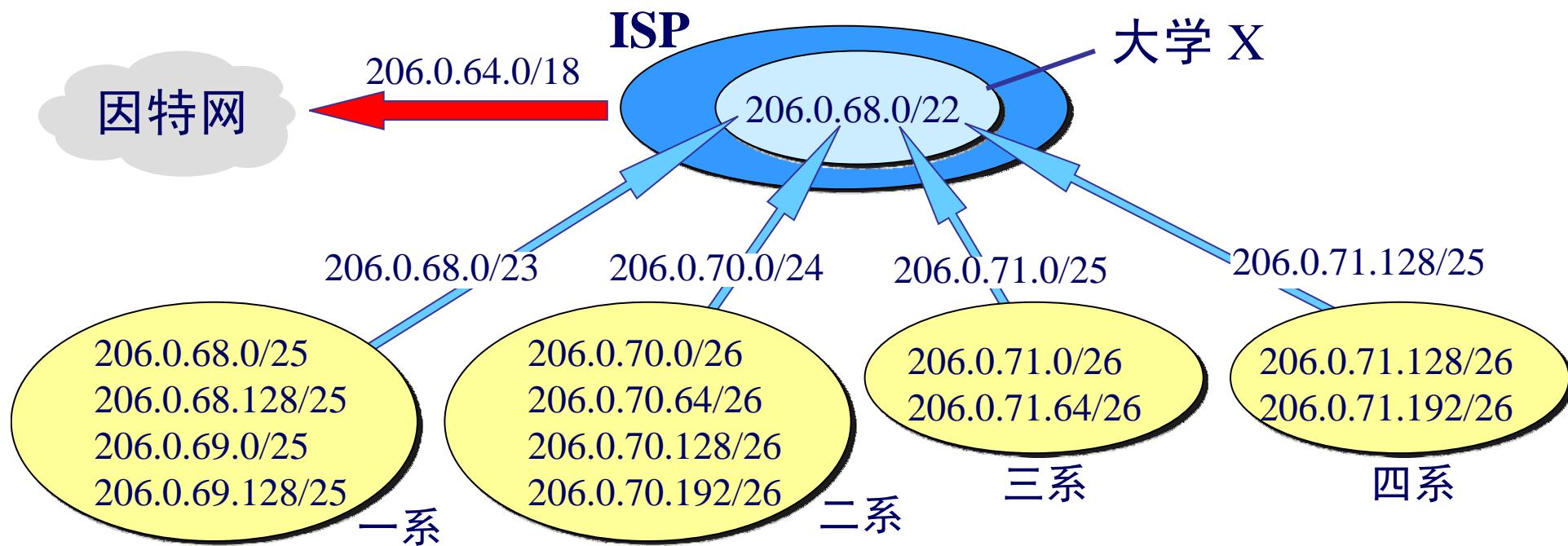
```
11111111 11000000 00000000 00000000
```
- 网络前缀的后面加一个星号 * 的表示方法
如 00001010 00*，在星号 * 之前是网络前缀，而星号 * 表示 IP 地址中的主机号，可以是任意值。



构成超网

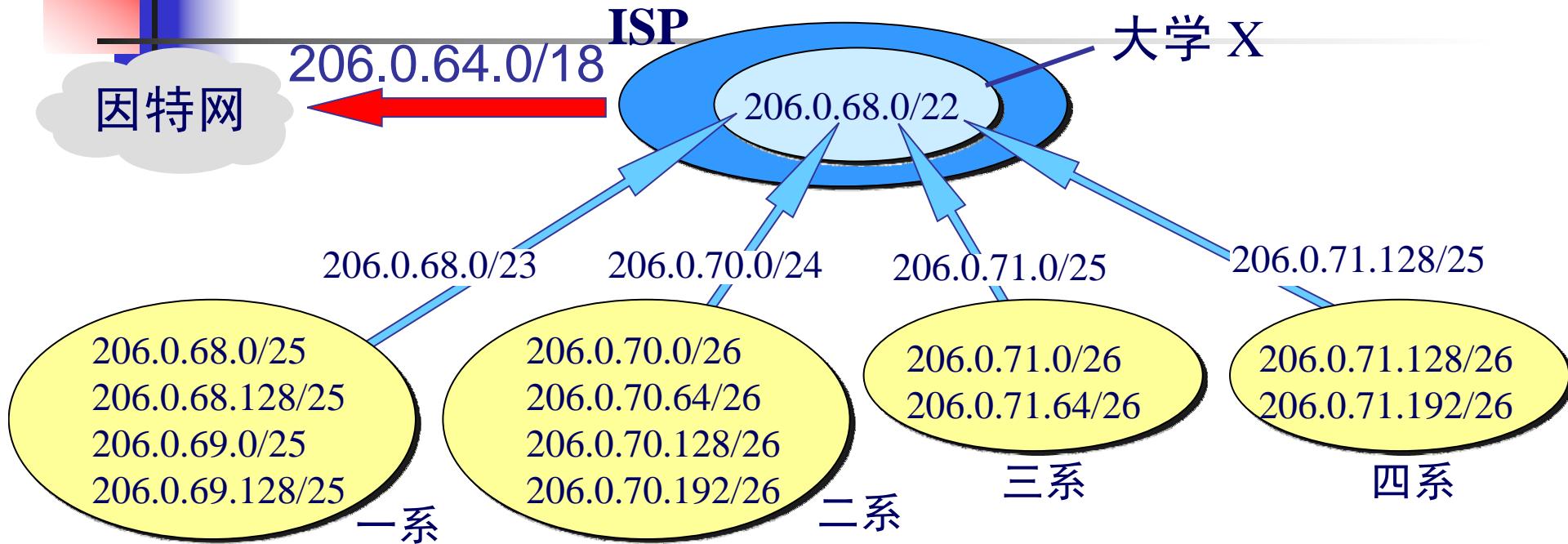
- 前缀长度不超过 23 位的 CIDR 地址块都包含了多个 C 类地址。
- 这些 C 类地址合起来就构成了超网。
- CIDR 地址块中的地址数一定是 2 的整数次幂。
- 网络前缀越短，其地址块所包含的地址数就越多。而在三级结构的IP地址中，划分子网是使网络前缀变长。

CIDR 地址块划分举例

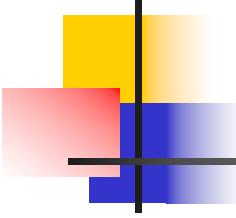


单位	地址块	二进制表示	地址数
ISP	206.0.64.0/18	11001110.00000000.01*	16384
大学	206.0.68.0/22	11001110.00000000.010001*	1024
一系	206.0.68.0/23	11001110.00000000.0100010*	512
二系	206.0.70.0/24	11001110.00000000.01000110.*	256
三系	206.0.71.0/25	11001110.00000000.01000111.0*	128
四系	206.0.71.128/25	11001110.00000000.01000111.1*	128

CIDR 地址块划分举例



这个 ISP 共有 64 个 C 类网络。如果不采用 CIDR 技术，则在与该 ISP 的路由器交换路由信息的每一个路由器的路由表中，就需要有 64 个项目。但采用地址聚合后，只需用路由聚合后的 1 个项目 206.0.64.0/18 就能找到该 ISP。



2. 最长前缀匹配

- 使用 CIDR 时，路由表中的每个项目由“网络前缀”和“下一跳地址”组成。在查找路由表时可能会得到不止一个匹配结果。
- 应当从匹配结果中选择具有最长网络前缀的路由：**最长前缀匹配**(longest-prefix matching)。
- 网络前缀越长，其地址块就越小，因而路由就越具体(more specific)。
- 最长前缀匹配又称为**最长匹配**或**最佳匹配**。

最长前缀匹配举例

收到的分组的目的地址 $D = 206.0.71.128$

路由表中的项目： $206.0.68.0/22$ (ISP)

$206.0.71.128/25$ (四系)

查找路由表中的第 1 个项目

第 1 个项目 $206.0.68.0/22$ 的掩码 M 有 22 个连续的 1。

$$M = 11111111 11111111 11111100 00000000$$

因此只需把 D 的第 3 个字节转换成二进制。

$$M = \boxed{11111111 11111111 11111100 00000000}$$

AND	$D =$	206.	0.	01000111.	0
<hr/>					
$\boxed{206. 0. 01000100. 0}$					

与 $206.0.68.0/22$ 匹配

最长前缀匹配举例

收到的分组的目的地址 $D = 206.0.71.128$

路由表中的项目： 206.0.68.0/22 (ISP)

206.0.71.128/25 (四系)

再查找路由表中的第 2 个项目

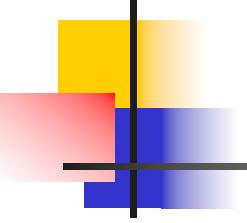
第 2 个项目 206.0.71.128/25 的掩码 M 有 25 个连续的 1。

$$M = 11111111 11111111 11111111 10000000$$

因此只需把 D 的第 4 个字节转换成二进制。

$$\begin{array}{rcl} M = & \boxed{11111111} & 11111111 11111111 10000000 \\ \text{AND } D = & \boxed{206.} & 0. 71. \boxed{10000000} \\ \hline & \boxed{206.} & 0. 71. \boxed{10000000} \end{array}$$

与 206.0.71.128/25 匹配

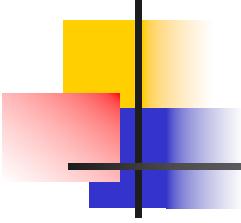


最长前缀匹配

$D \text{ AND } (11111111\ 11111111\ 11111100\ 00000000)$
= 206.0.68.0/22 匹配

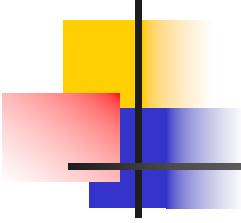
$D \text{ AND } (11111111\ 11111111\ 11111111\ 10000000)$
= 206.0.71.128/25 匹配

- 选择两个匹配的地址中更具体的一个，即选择**最长前缀的地址**。



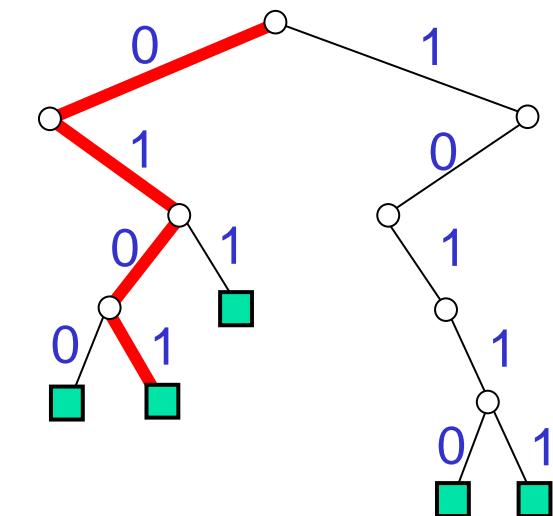
3. 使用二叉线索查找路由表

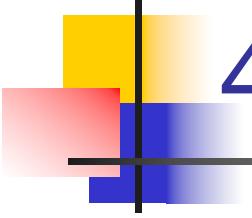
- 当路由表的项目数很大时，怎样设法减小路由表的查找时间就成为一个非常重要的问题。
- 为了进行更加有效的查找，通常是将无分类编址的路由表存放在一种层次的数据结构中，然后自上而下地按层次进行查找。这里最常用的就是**二叉线索**(binary trie)。
- IP 地址中从左到右的比特值决定了从根结点逐层向下层延伸的路径，而二叉线索中的各个路径就代表路由表中存放的各个地址。
- 为了提高二叉线索的查找速度，广泛使用了各种压缩技术。



用 5 个前缀构成的二叉线索

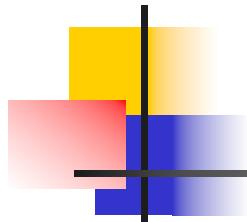
32 位的 IP 地址	唯一前缀
01000110 00000000 00000000 00000000	0100
01010110 00000000 00000000 00000000	0101
01100001 00000000 00000000 00000000	011
10110000 00000010 00000000 00000000	10110
10111011 00001010 00000000 00000000	10111





4.4 网际控制报文协议 ICMP

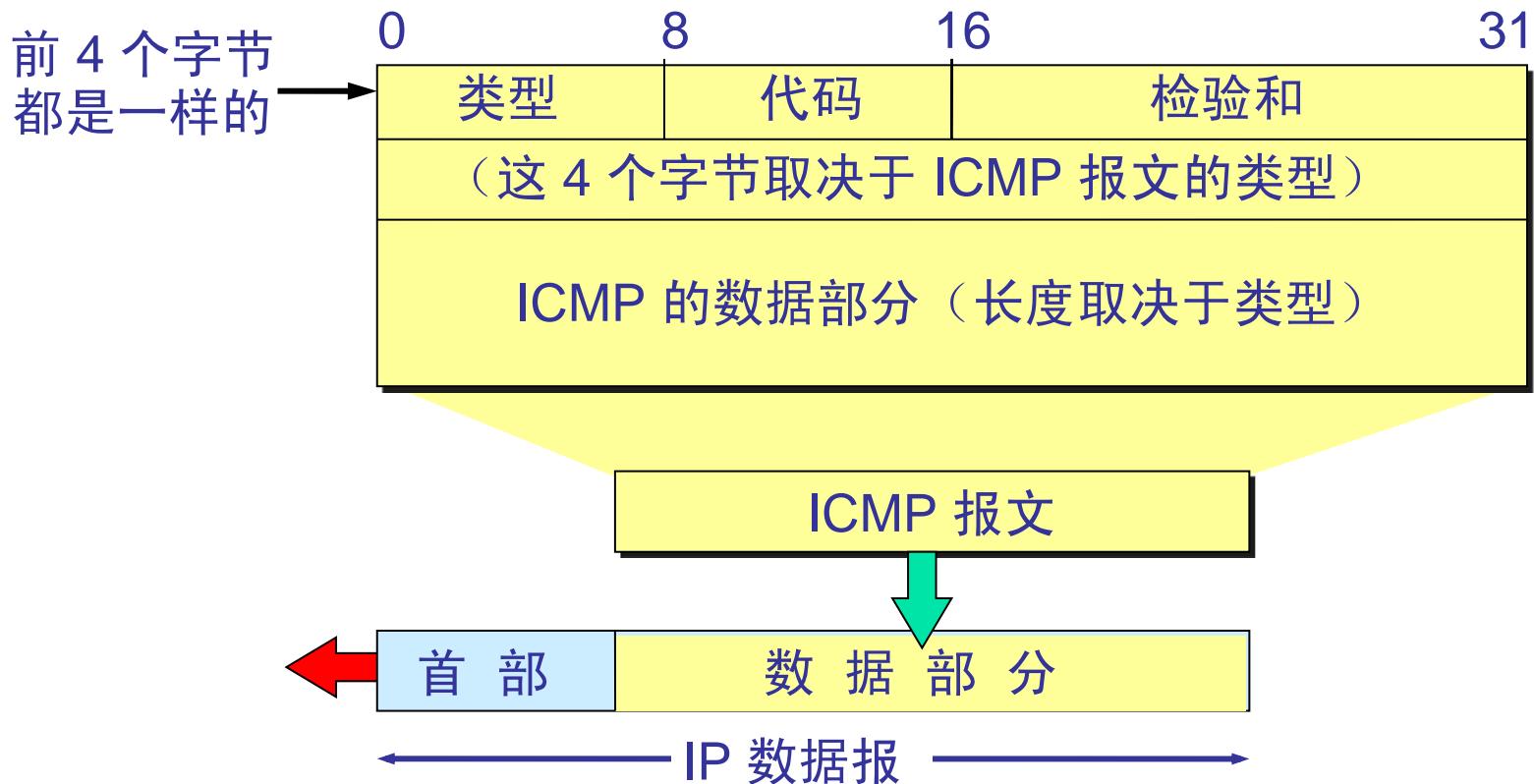
- 为了提高 IP 数据报交付成功的机会，在网际层使用了网际控制报文协议 ICMP (Internet Control Message Protocol)。
- ICMP 允许主机或路由器报告差错情况和提供有关异常情况的报告。
- ICMP 不是高层协议，而是 IP 层的协议。
- ICMP 报文作为 IP 层数据报的数据，加上数据报的头部，组成 IP 数据报发送出去。

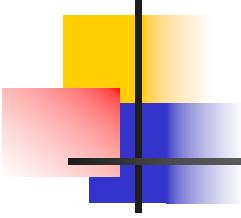


4.4.1 ICMP 报文的种类

- ICMP 报文的种类有两种，即 ICMP 差错报告报文和 ICMP 询问报文。
- ICMP 报文的前 4 个字节是统一的格式，共有三个字段：即类型、代码和检验和。接着的 4 个字节的内容与 ICMP 的类型有关。

ICMP 报文的格式

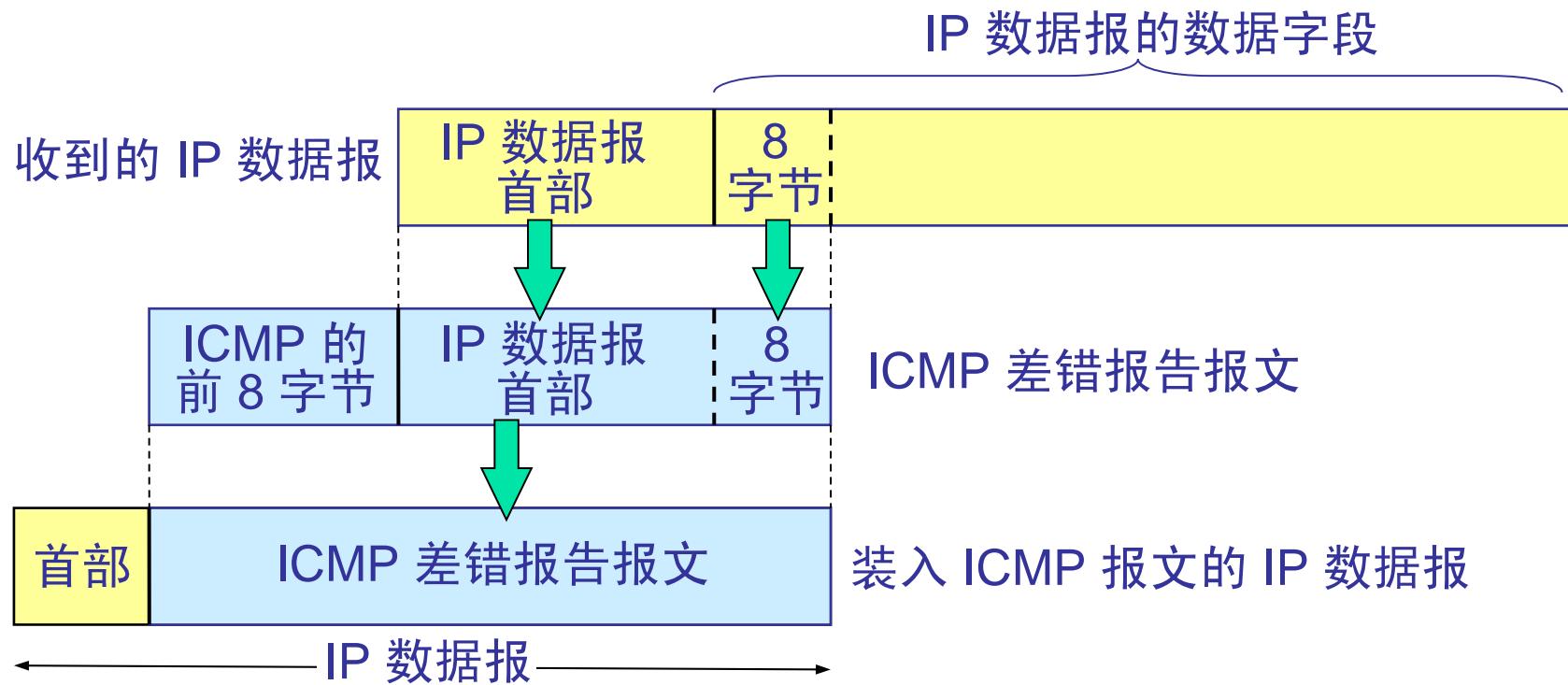


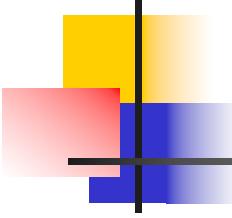


ICMP 差错报告报文共有 5 种

- 终点不可达
- 源点抑制(Source quench)
- 时间超过
- 参数问题
- 改变路由（重定向）(Redirect)

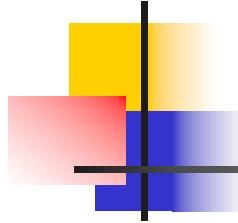
ICMP 差错报告报文的数据字段的内容





不应发送 ICMP 差错报告报文的几种情况

- 对 ICMP 差错报告报文不再发送 ICMP 差错报告报文。
- 对第一个分片的数据报片的所有后续数据报片都不发送 ICMP 差错报告报文。
- 对具有多播地址的数据报都不发送 ICMP 差错报告报文。
- 对具有特殊地址（如 127.0.0.0 或 0.0.0.0）的数据报不发送 ICMP 差错报告报文。

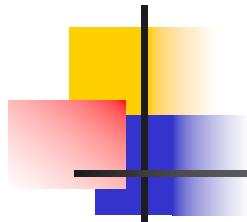


ICMP 询问报文有两种

- 回送请求和回答报文
- 时间戳请求和回答报文

下面的几种 ICMP 报文不再使用

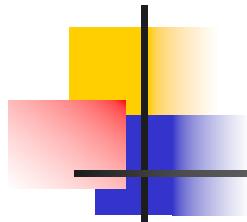
- 信息请求与回答报文
- 掩码地址请求和回答报文
- 路由器询问和通告报文



4.4.2 ICMP的应用举例

PING (Packet InterNet Groper)

- PING 用来测试两个主机之间的连通性。
- PING 使用了 ICMP 回送请求与回送回答报文。
- PING 是应用层直接使用网络层 ICMP 的例子，它没有通过运输层的 TCP 或 UDP。



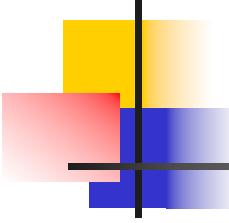
PING 的应用举例

```
C:\Documents and Settings\XXX>ping mail.sina.com.cn

Pinging mail.sina.com.cn [202.108.43.230] with 32 bytes of data:

Reply from 202.108.43.230: bytes=32 time=368ms TTL=242
Reply from 202.108.43.230: bytes=32 time=374ms TTL=242
Request timed out.
Reply from 202.108.43.230: bytes=32 time=374ms TTL=242

Ping statistics for 202.108.43.230:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 368ms, Maximum = 374ms, Average = 372ms
```



Traceroute 的应用举例

```
C:\Documents and Settings\XXX>tracert mail.sina.com.cn

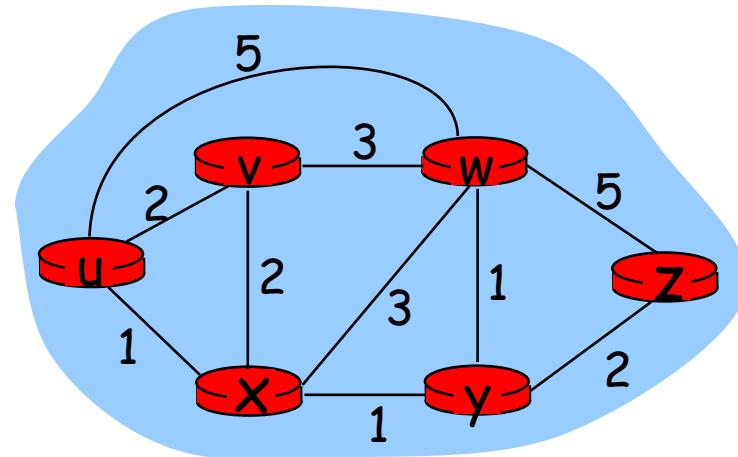
Tracing route to mail.sina.com.cn [202.108.43.230]
over a maximum of 30 hops:

 1  24 ms    24 ms    23 ms  222.95.172.1
 2  23 ms    24 ms    22 ms  221.231.204.129
 3  23 ms    22 ms    23 ms  221.231.206.9
 4  24 ms    23 ms    24 ms  202.97.27.37
 5  22 ms    23 ms    24 ms  202.97.41.226
 6  28 ms    28 ms    28 ms  202.97.35.25
 7  50 ms    50 ms    51 ms  202.97.36.86
 8  308 ms   311 ms   310 ms  219.158.32.1
 9  307 ms   305 ms   305 ms  219.158.13.17
10  164 ms   164 ms   165 ms  202.96.12.154
11  322 ms   320 ms   2988 ms  61.135.148.50
12  321 ms   322 ms   320 ms  freemail43-230.sina.com [202.108.43.230]

Trace complete.
```

4.5 因特网的路由选择协议

- 图抽象



Graph: $G = (N, E)$

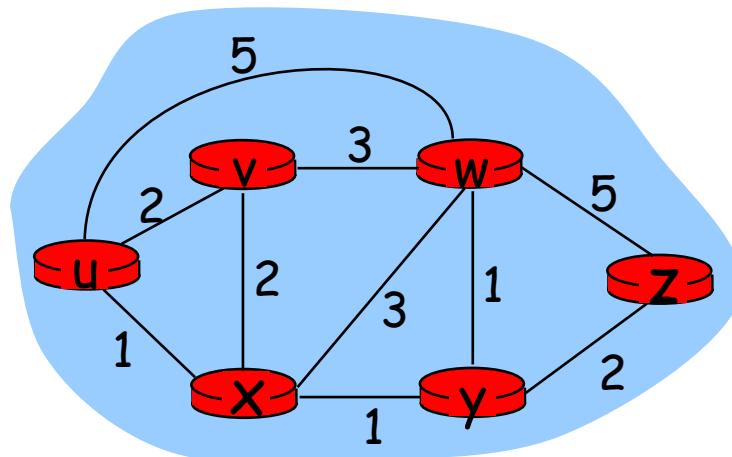
$N = \text{set of routers} = \{ u, v, w, x, y, z \}$

$E = \text{set of links} = \{ (u,v), (u,x), (v,x), (v,w), (x,w), (x,y), (w,y), (w,z), (y,z) \}$

图抽象也用于其他的网络分析场合，

如: P2P网络, N 代表结点集; E 代表结点之间的TCP 链接结合。

图抽象: 代价

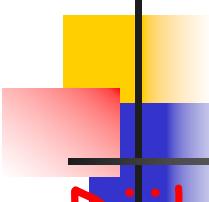


- $c(x, x') =$ 链路 (x, x') 的代价
 - e.g., $c(w, z) = 5$
- 代价可能为 1
- 或者与带宽成反比
- 或者与链路的拥塞程度成反比

路径代价 $\text{Cost}(x_1, x_2, x_3, \dots, x_p) = c(x_1, x_2) + c(x_2, x_3) + \dots + c(x_{p-1}, x_p)$

Question: u和z之间的最小代价路径是哪一条?

路由算法：找到最小代价路径的算法

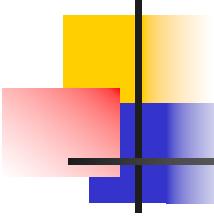


集中式的路由选择策略

Dijkstra's algorithm

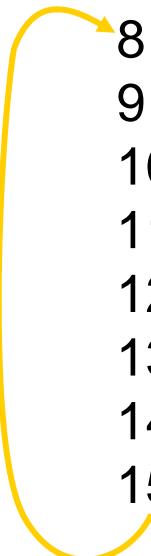
- 所有节点均了解网络的拓扑和链路代价
 - 所有节点拥有同样的路由信息
 - 全局信息的获得方式：链路状态 (**link State**) 广播
- 每一个节点均计算到所有其他节点的最短路径
 - 形成该结点的**转发表**
 - 迭代计算： K 次循环，求出到 K 个节点的最短路径

- 符号
- $c(x,y)$: 链路(x, y)的代价; $= \infty$ if 非直接相连
- $D(v)$: 从源结点到目的结点 v 当前迭代后的路径代价
- $p(v)$: 从源结点到目的结点 v 当前迭代后的路径上 v 节点前的节点
- N' : 已经计算出最短路径的节点集



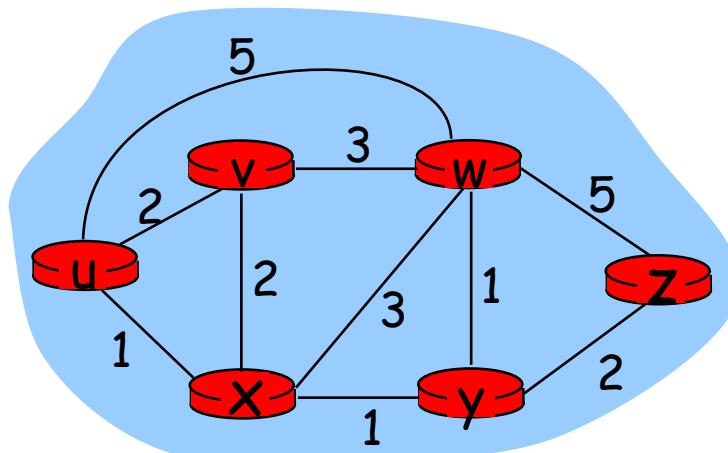
Dijkstra's Algorithm

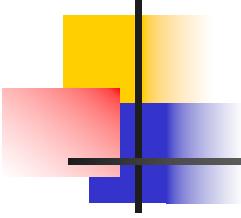
```
1 Initialization:
2    $N' = \{u\}$ 
3   for all nodes  $v$ 
4     if  $v$  adjacent to  $u$ 
5       then  $D(v) = c(u,v)$ 
6     else  $D(v) = \infty$ 
7
8 Loop
9   find  $w$  not in  $N'$  such that  $D(w)$  is a minimum
10  add  $w$  to  $N'$ 
11  update  $D(v)$  for all  $v$  adjacent to  $w$  and not in  $N'$  :
12     $D(v) = \min( D(v), D(w) + c(w,v) )$ 
13  /* new cost to  $v$  is either old cost to  $v$  or known
14    shortest path cost to  $w$  plus cost from  $w$  to  $v$  */
15 until all nodes in  $N'$ 
```



Dijkstra's algorithm 举例

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					

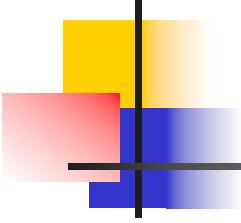




4.5.1 有关路由选择协议的基本概念

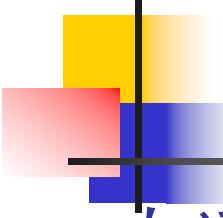
■ 理想的路由算法

- 算法必须是正确的。
- 算法在计算上应简单。
- 算法应能适应通信量和网络拓扑的变化，即自适应性。
- 算法应具有稳定性。
- 算法应是公平的。
- 算法应是最佳的。



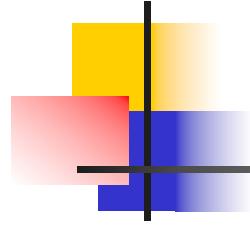
关于“最佳路由”

- 不存在一种绝对的最佳路由算法。
- 所谓“最佳”只能是相对于某一种特定要求下得出的较为合理的选择而已。
- 实际的路由选择算法，应尽可能接近于理想的算法。
- 路由选择是个非常复杂的问题
 - 是网络中的所有结点共同协调工作的结果。
 - 路由选择的环境往往是不断变化的，而这种变化有时无法事先知道。



路由算法分类

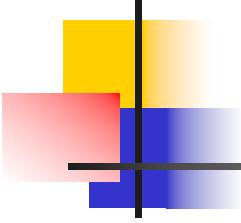
- 自适应性分类
- 静态路由选择策略
 - 非自适应路由选择，路径不能及时适应网络状态的变化
 - 简单、开销较小
- 动态路由选择策略
 - 自适应路由选择，路径随网络状态的变化
 - 定期更新链路代价信息
 - 路径随着链路状态改变
 - 实现开销大
- 信息程度分类
- 集中式路由选择策略
 - 所有路由器拥有全部拓扑和链路状态信息
 - “link state” algorithms
- 分布式路由选择策略
 - 路由器仅了解其物理链路的邻居情况，及相关链路的代价
 - 通过反复和邻居结点进行路由信息交换，迭代地进行路由计算
 - “distance vector” algorithms



4.5.3 内部网关协议 OSPF (Open Shortest Path First)

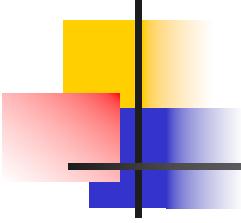
OSPF 协议的基本特点

- “开放”表明 OSPF 协议不是受某一家厂商控制，而是公开发表的。
- “最短路径优先”是因为使用了 Dijkstra 提出的最短路径算法 SPF
- OSPF 只是一个协议的名字，它并不表示其他的路由选择协议不是“最短路径优先”。
- 是**分布式的链路状态协议**。



三个要点

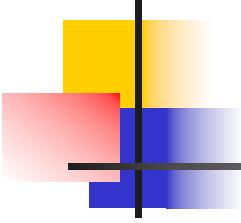
- 向本自治系统中所有路由器发送信息，这里使用的方法是洪泛法。
- 发送的信息就是与本路由器相邻的所有路由器的链路状态，但这只是路由器所知道的部分信息。
 - “链路状态”就是说明本路由器都和哪些路由器相邻，以及该链路的“度量”(metric)。
- 只有当链路状态发生变化时，路由器才用洪泛法向所有路由器发送此信息。



链路状态数据库

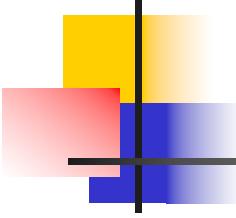
(link-state database)

- 由于各路由器之间频繁地交换链路状态信息，因此所有的路由器最终都能建立一个链路状态数据库。
- 这个数据库实际上就是**全网的拓扑结构图**，它在全网范围内是一致的（这称为链路状态数据库的同步）。
- OSPF 的链路状态数据库能较快地进行更新，使各个路由器能及时更新其路由表。OSPF 的更新过程收敛得快是其重要优点。



OSPF 直接用 IP 数据报传送

- OSPF 不用 UDP 而是直接用 IP 数据报传送。
- OSPF 构成的数据报很短
 - 减少路由信息的通信量。
 - 不必将长的数据报分片。分片传送的数据报只要丢失一个，就无法组装成原来的数据报，而整个数据报就必须重传。



OSPF 的其他特点

- OSPF 对不同的链路可根据 IP 分组的不同服务类型 TOS 而设置成不同的代价。因此，OSPF 对于不同类型的业务可计算出不同的路由。
- 如果到同一个目的网络有多条相同代价的路径，那么可以将通信量分配给这几条路径。这叫作多路径间的负载平衡。
- 所有在 OSPF 路由器之间交换的分组都具有鉴别的功能。
- 支持可变长度的子网划分和无分类编址 CIDR。
- 每一个链路状态都带上一个 32 位的序号，序号越大状态就越新。

OSPF 分组

位

0

8

16

31



24 字节

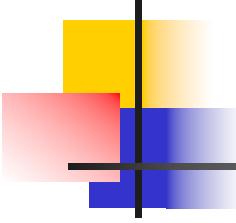
OSPF 分组首部

类型 1 至类型 5 的 OSPF 分组

IP数据报首部

OSPF 分组

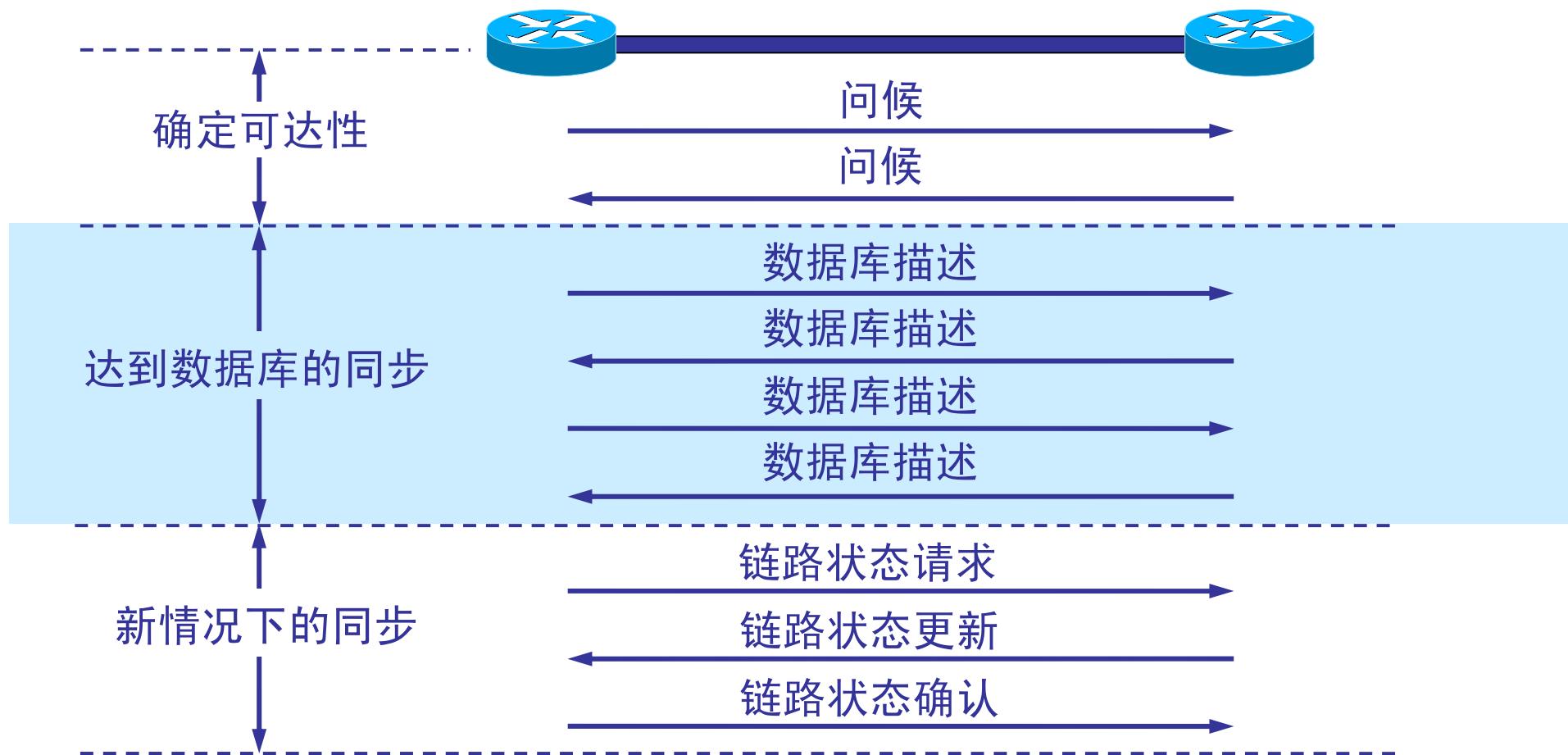
IP 数据报



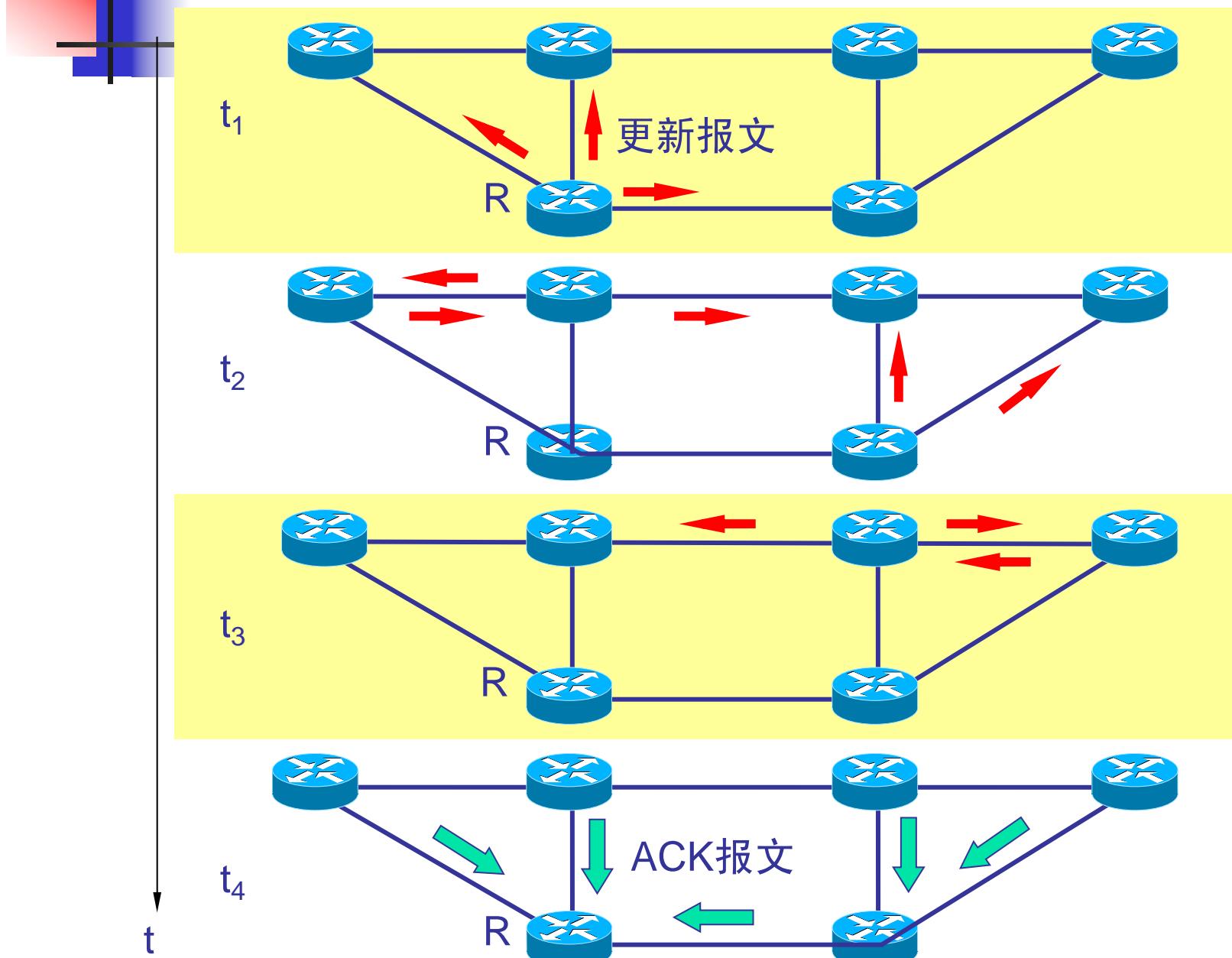
2. OSPF 的五种分组类型

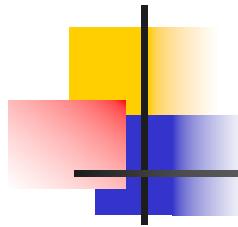
- 类型1， 问候(Hello)分组。
- 类型2， 数据库描述(Database Description)分组。
- 类型3， 链路状态请求(Link State Request)分组。
- 类型4， 链路状态更新(Link State Update)分组，
用洪泛法对全网更新链路状态。
- 类型5， 链路状态确认(Link State Acknowledgment)
分组。

OSPF的基本操作



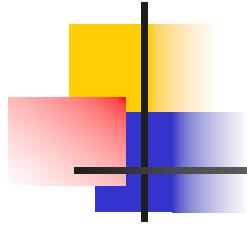
OSPF 使用的是可靠的洪泛法





OSPF 的其他特点

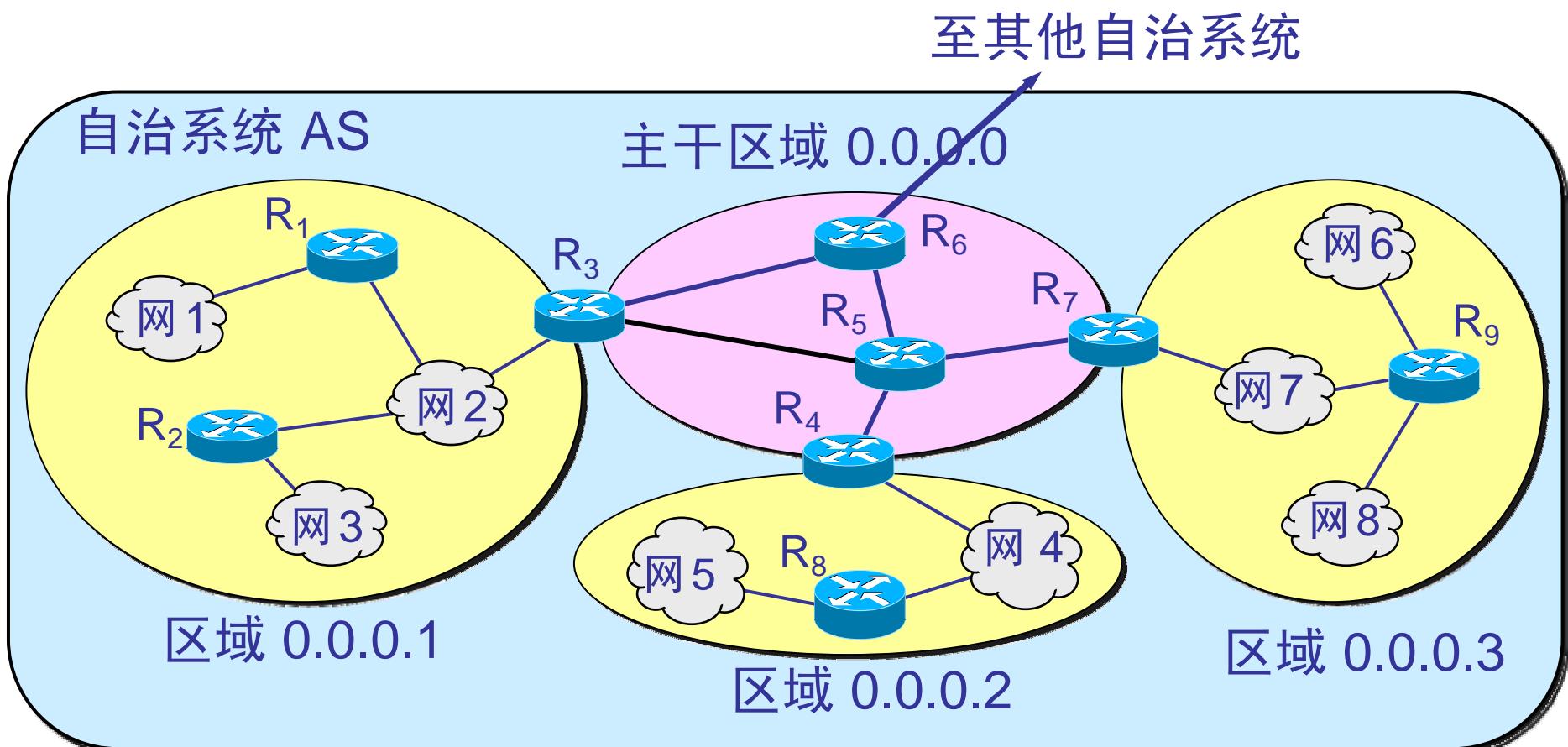
- OSPF 还规定每隔一段时间，如 30 分钟，要刷新一次数据库中的链路状态。
- 由于一个路由器的链路状态只涉及到与相邻路由器的连通状态，因而与整个互联网的规模并无直接关系。因此当互联网规模很大时，OSPF 协议要比距离向量协议 RIP 好得多。
- OSPF 没有“坏消息传播得慢”的问题，据统计，其响应网络变化的时间小于 100 ms。

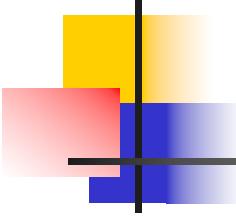


OSPF 的区域(area)

- 为了使 OSPF 能够用于规模很大的网络，OSPF 将一个自治系统再划分为若干个更小的范围，叫作**区域**。
- 每一个区域都有一个 32 位的区域标识符（用点分十进制表示）。
- 区域也不能太大，在一个区域内的路由器最好不超过 200 个。

OSPF 划分为两种不同的区域

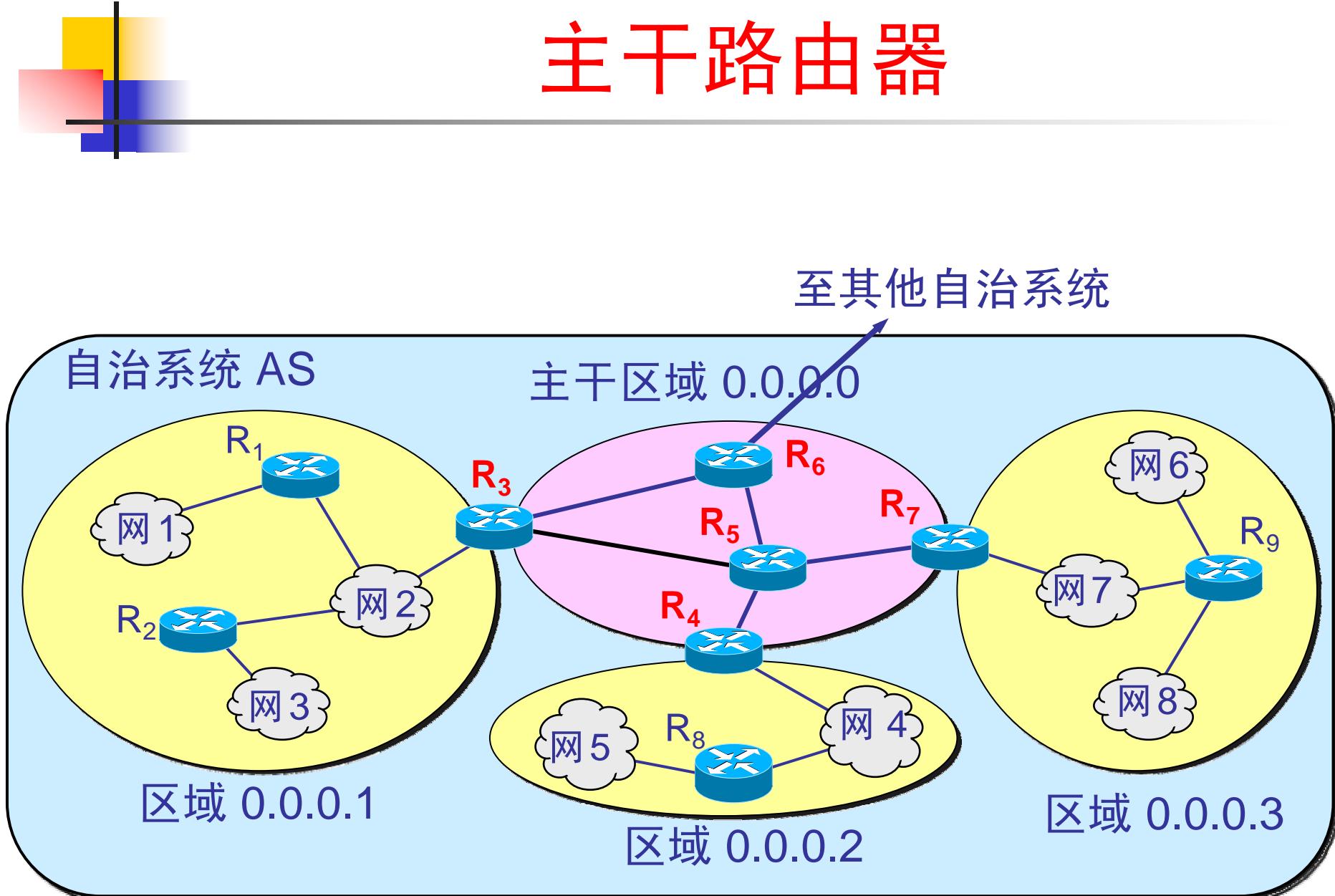




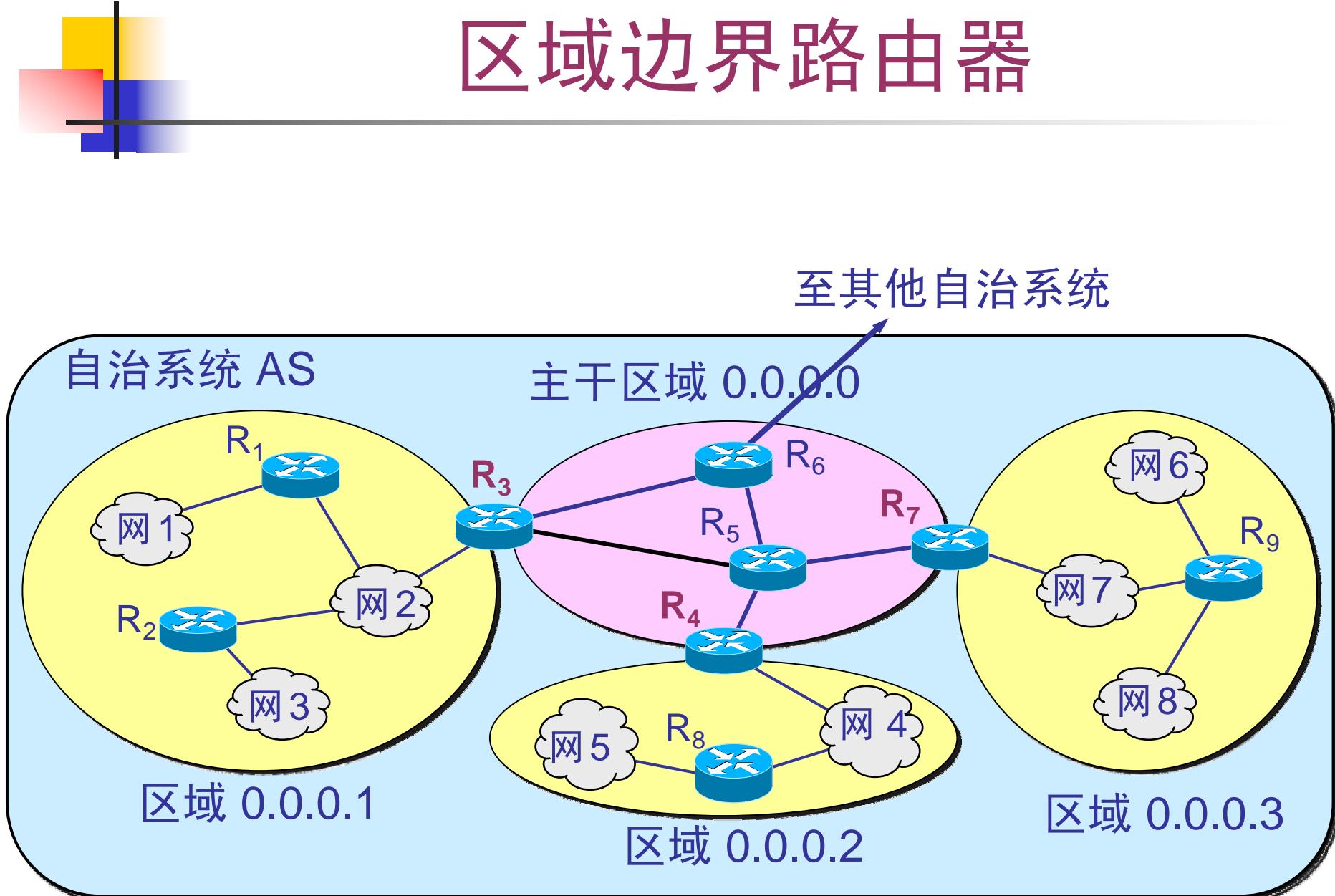
划分区域

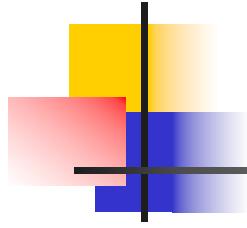
- 划分区域的好处就是将利用洪泛法交换链路状态信息的范围局限于每一个区域而不是整个的自治系统，这就减少了整个网络上的通信量。
- 在一个区域内部的路由器只知道本区域的完整网络拓扑，而不知道其他区域的网络拓扑的情况。
- OSPF 使用层次结构的区域划分。在上层的区域叫作**主干区域**(backbone area)。主干区域的标识符规定为0.0.0.0。主干区域的作用是用来连通其他在下层的区域。

主干路由器



区域边界路由器

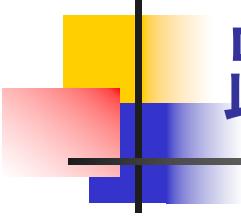




4.5.2 内部网关协议 RIP (Routing Information Protocol)

工作原理

- RIP 是一种分布式的基于距离向量的路由选择协议。
- RIP 协议要求网络中的每一个路由器都要维护从它自己到其他每一个目的网络的距离记录。



距离向量 (DV) 算法理论基础

Bellman-Ford Equation

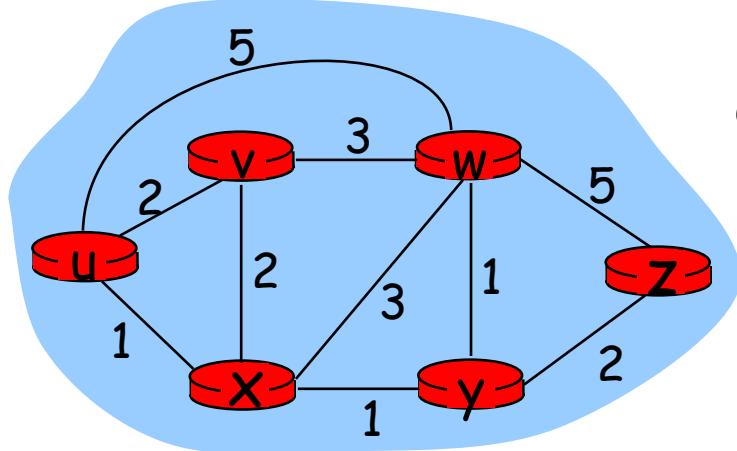
定义

$d_x(y) :=$ 从 x 到 y 的代价最小路径的代价

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

Min运算是包括 x 的所有邻居。

Bellman-Ford 举例

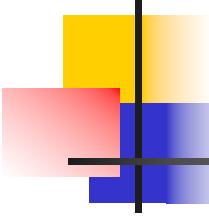


$$d_v(z) = 5, d_x(z) = 3, d_w(z) = 3$$

B-F equation 指出：

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

实现最短路径的结点是从邻居结点中选择的下一跳
 转发表



DV算法

- 每个结点定期向所有邻居发送距离向量表
- 当结点从其邻居处获得距离向量表后，该结点根据 B-F equation计算自己新的距离向量表

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

- 当结点规模有限时，所有结点的距离向量表均会收敛到最短路径的距离向量。

DV特点

迭代、异步：

每次本地更新由以下情况

引起

- 本地链路代价改变
- 收到来自邻居的距离向量表更新消息

分布式：

- 每个节点仅在自身距离向量表发生改变时通知邻居
 - 被通知的邻居结点再根据自身距离向量表的更新结果决定是否继续通知他的邻居

节点：

等待 邻居发送新的距离向量表

重新计算 自身距离向量表

若到任意目的网络的距离向量变化，通知 所有邻居

node x table

cost to

	x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

$$D_x(y) = \min\{c(x,y) + D_y(y), c(x,z) + D_z(y)\}$$

$$= \min\{2+0, 7+1\} = 2$$

$$D_x(z) = \min\{c(x,y) + D_y(z), c(x,z) + D_z(z)\}$$

$$= \min\{2+1, 7+0\} = 3$$

node y table

cost to

	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

node z table

cost to

	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0

cost to

	x	y	z
x	0	2	3
y	2	0	1
z	7	1	0

cost to

	x	y	z
x	0	2	7
y	2	0	1
z	7	1	0

cost to

	x	y	z
x	0	2	7
y	2	0	1
z	3	1	0

cost to

cost to

	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

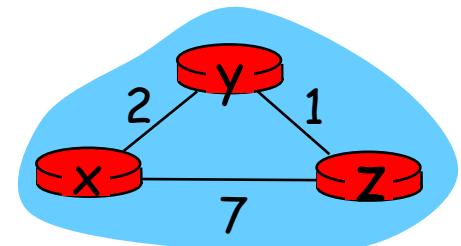
cost to

	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

cost to

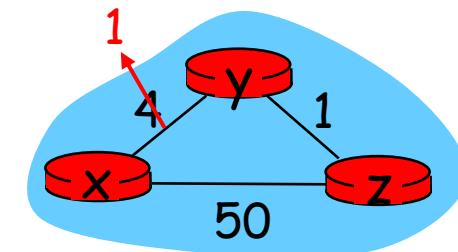
	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

→ time



DV: 链路代价改变

- 结点发现本地链路代价改变
- 更新路由信息，重新计算距离向量
- 如果引起距离向量改变，通知所有邻居

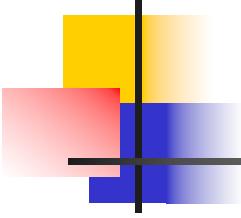


$t_{0:}$ y 发现本地链路代价改变，重新计算距离向量 $D_x(y)=1$, 通知所有邻居

“好消息传得快！”

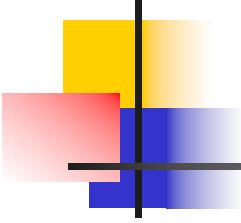
$t_1:$ z 收到来自 y 的新的距离向量表，重新计算自身的距离向量表，更新 $D_z(y)=2$, 通知所有邻居

$t_2:$ y 收到 z 更新后的距离向量表，重新计算自身的距离向量表，发现没有改动， y 不再发送新的距离向量表。



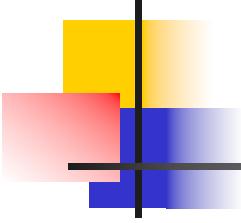
“距离”的定义

- 从一路由器到**直接连接**的网络的距离定义为 1。
- 从一个路由器到非直接连接的网络的距离定义为所经过的路由器数加 1。
- RIP 协议中的“距离”也称为“**跳数**”(hop count)，因为每经过一个路由器，跳数就加 1。
- 这里的“距离”实际上指的是“**最短距离**”。



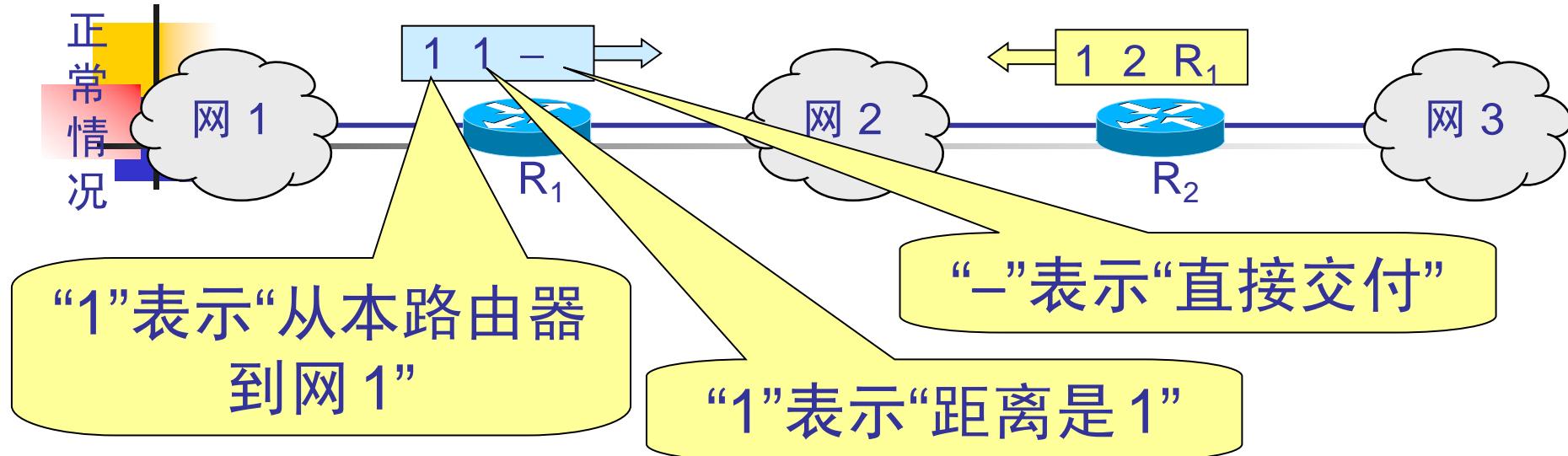
“距离”的定义

- RIP 认为一个好的路由就是它通过的路由器的数目少，即“距离短”。
- RIP 允许一条路径最多只能包含 15 个路由器。
- “距离”的最大值为 16 时即相当于不可达。可见 RIP 只适用于小型互联网。
- RIP 不能在两个网络之间同时使用多条路由。RIP 选择一个具有最少路由器的路由（即最短路由），哪怕还存在另一条高速(低时延)但路由器较多的路由。

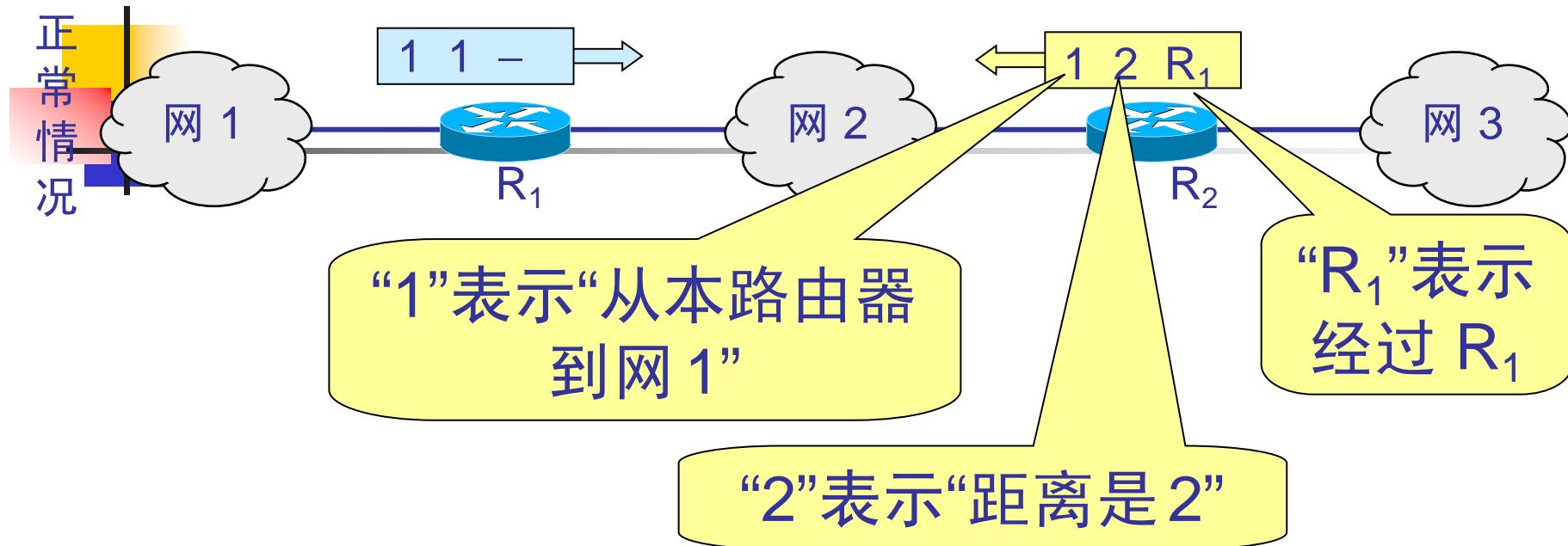


RIP 协议的三个要点

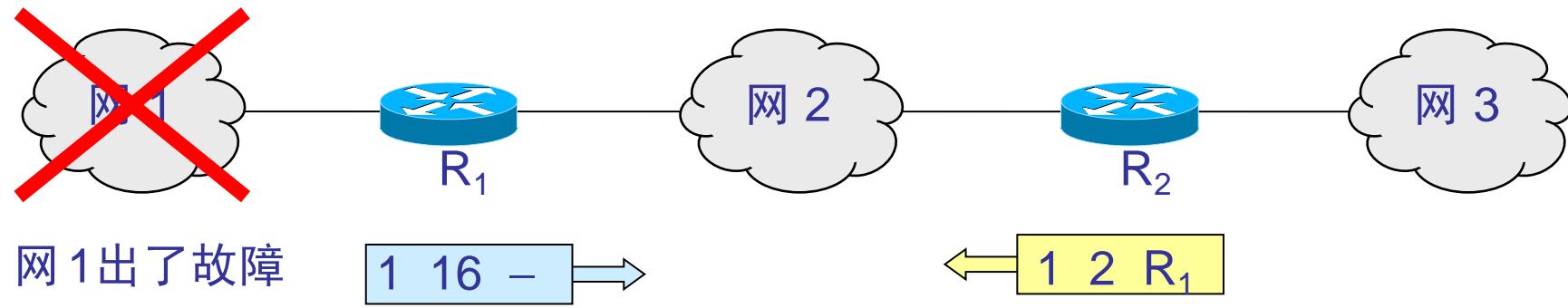
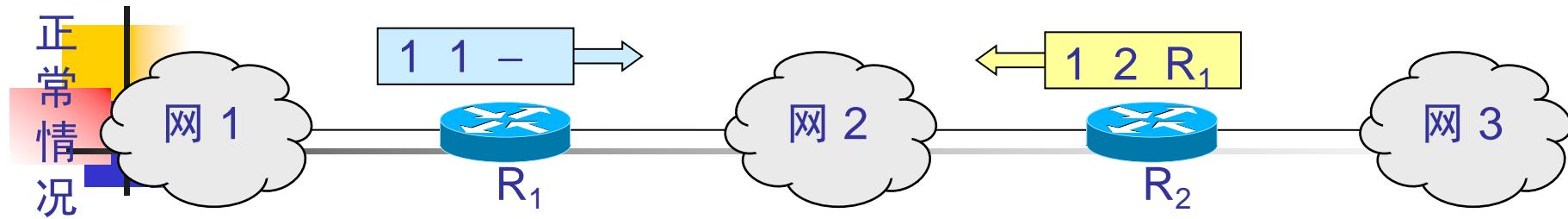
- 仅和相邻路由器交换信息。
- 交换的信息是当前本路由器所知道的全部信息，即自己的路由表。
- 按固定的时间间隔交换路由信息，例如每隔 30 秒。



R₁ 说：“我到网 1 的距离是 1， 是直接交付。”

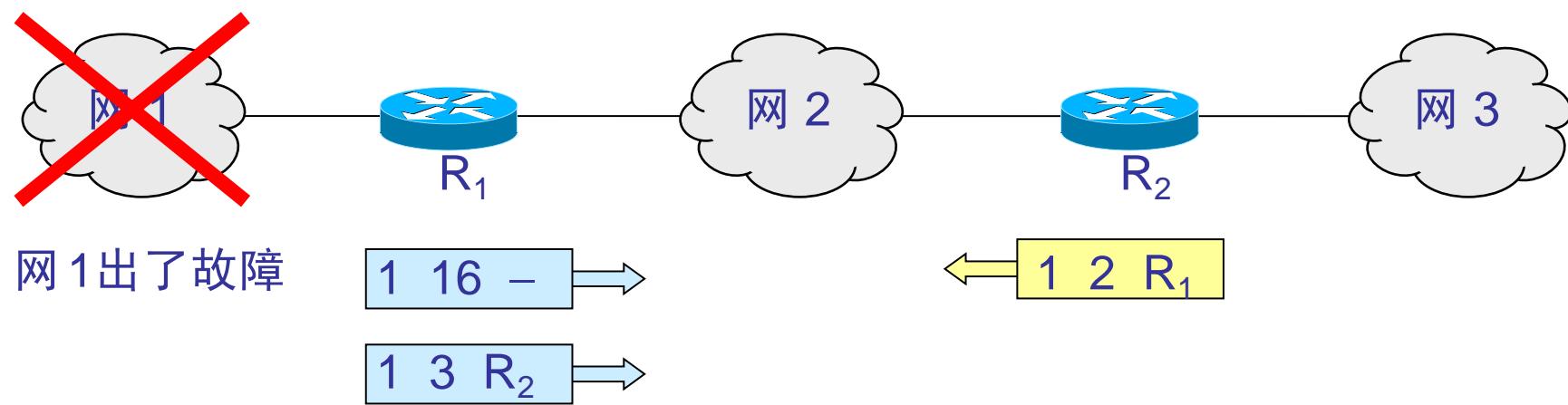
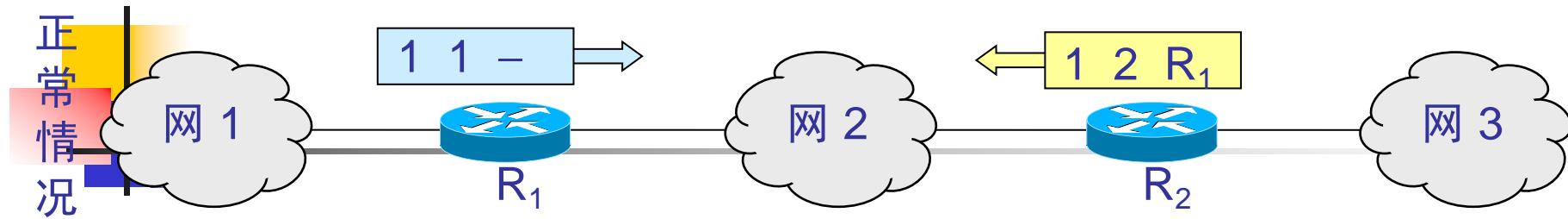


R₂ 说：“我到网 1 的距离是 2， 是经过 R₁。”

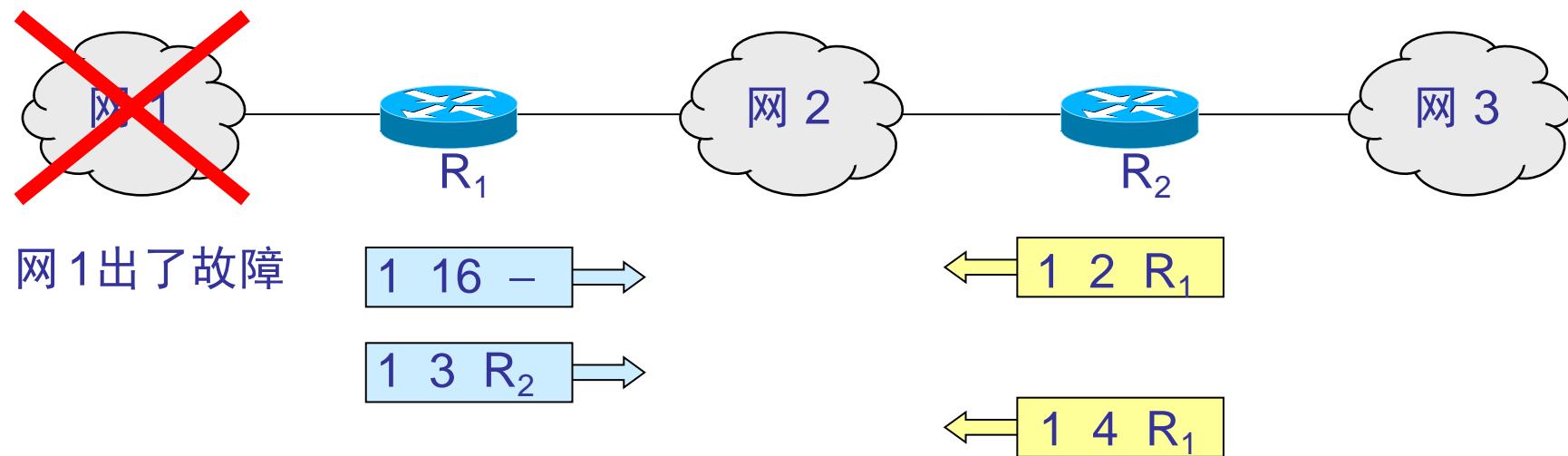
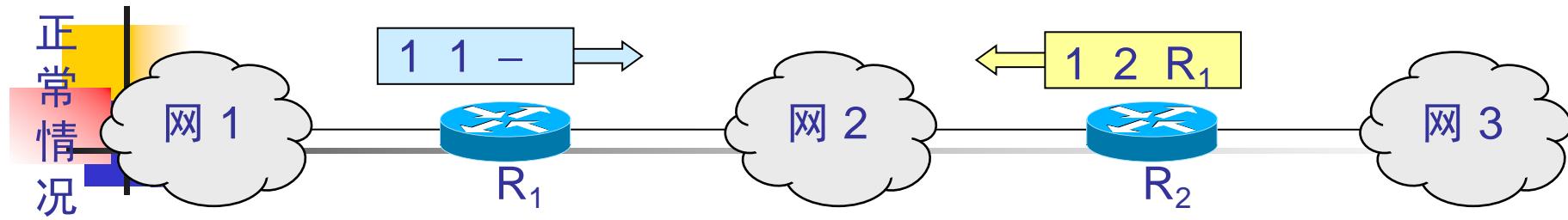


R₁ 说：“我到网 1 的距离是 16（表示无法到达），
是直接交付。”

但 R₂ 在收到 R₁ 的更新报文之前，还发送原来的报文，
因为这时 R₂ 并不知道 R₁ 出了故障。

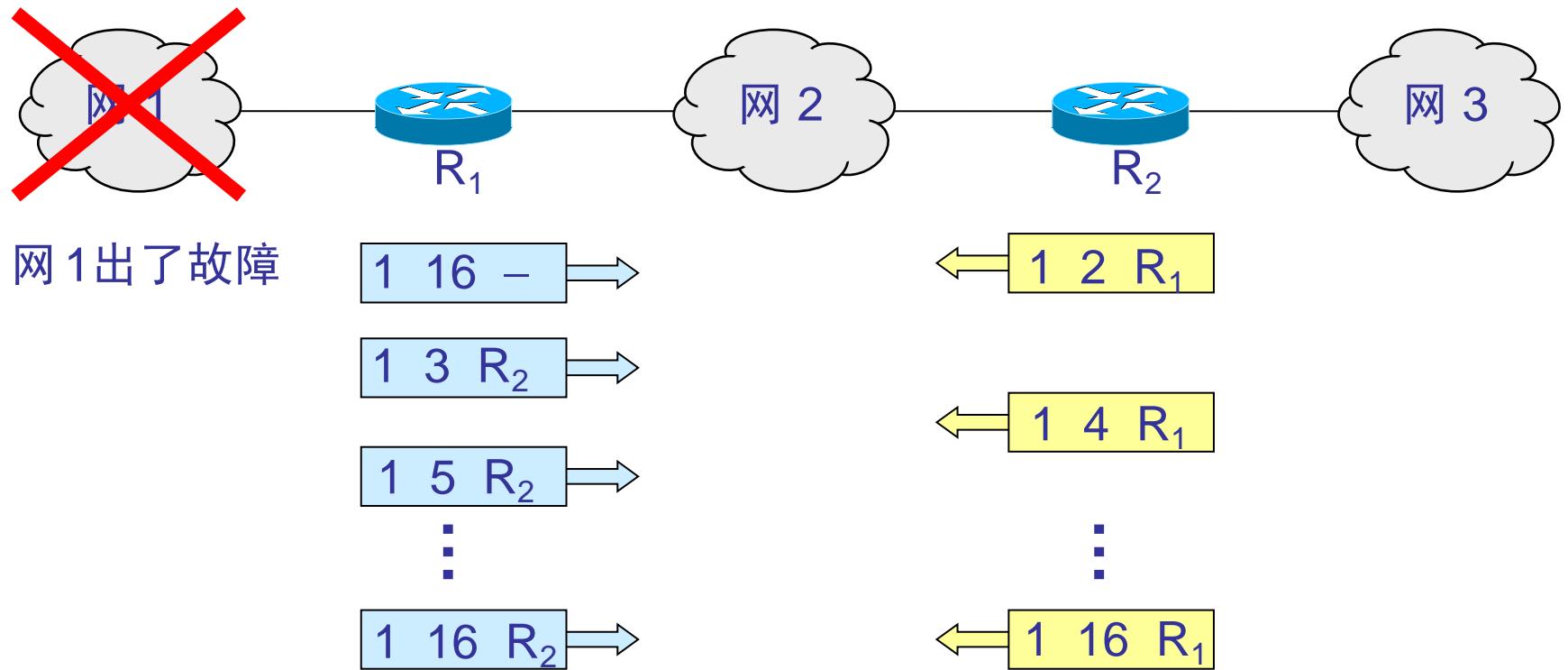


R₁ 收到 R₂ 的更新报文后，误认为可经过 R₂ 到达网 1，于是更新自己的路由表，说：“我到网 1 的距离是 3，下一跳经过 R₂”。然后将此更新信息发送给 R₂。

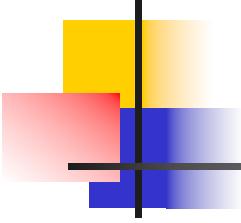


R₂ 以后又更新自己的路由表为“1, 4, R₁”，表明“我到网 1 距离是 4，下一跳经过 R₁”。

这就是好消息传播得快，而坏消息传播得慢。网络出故障的传播时间往往需要较长的时间(例如数分钟)。这是 RIP 的一个主要缺点。



这样不断更新下去，直到 R_1 和 R_2 到网 1 的距离都增大到 16 时， R_1 和 R_2 才知道网 1 是不可达的。



路由表的建立

- 路由器在刚刚开始工作时，只知道到直接连接的网络的距离（此距离定义为1）。
- 以后，每一个路由器也只和数目非常有限的相邻路由器交换并更新路由信息。
- 经过若干次更新后，所有的路由器最终都会知道到达本自治系统中任何一个网络的最短距离和下一跳路由器的地址。
- RIP 协议的收敛(convergence)过程较快，即在自治系统中所有的结点都得到正确的路由选择信息的过程。

2. 距离向量算法

收到相邻路由器（其地址为 X）的一个 RIP 报文：

(1) 先修改此 RIP 报文中的所有项目：把“下一跳”字段中的地址都改为 X，并把所有的“距离”字段的值加 1。

(2) 对修改后的 RIP 报文中的每一个项目，重复以下步骤：

若项目中的目的网络不在路由表中，则把该项目加到路由表中。

否则

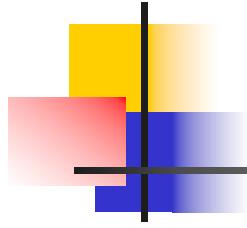
若下一跳字段给出的路由器地址是同样的，则把收到的项目替换原路由表中的项目。

否则

若收到项目中的距离小于路由表中的距离，则进行更新，否则，什么也不做。

(3) 若 3 分钟还没有收到相邻路由器的更新路由表，则把此相邻路由器记为不可达路由器，即将距离置为 16（距离为 16 表示不可达）。

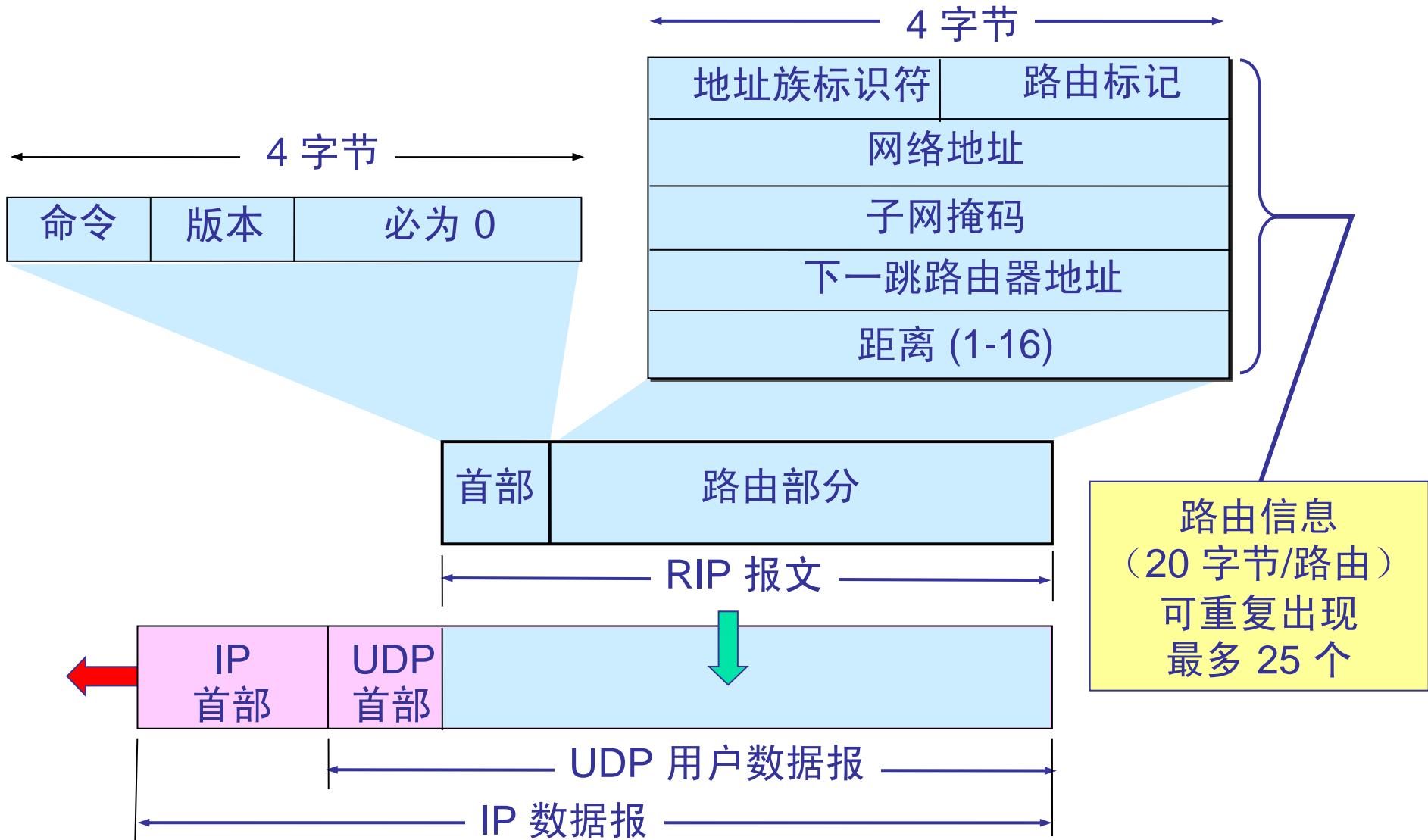
(4) 返回。

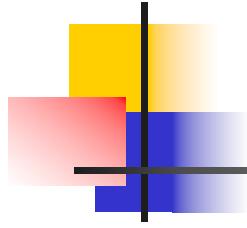


路由器之间交换信息

- RIP协议让互联网中的所有路由器都和自己的相邻路由器不断交换路由信息，并不断更新其路由表，使得从每一个路由器到每一个目的网络的路由都是最短的（即跳数最少）。
- 虽然所有的路由器最终都拥有了整个自治系统的全局路由信息，但由于每一个路由器的位置不同，它们的路由表当然也应当是不同的。

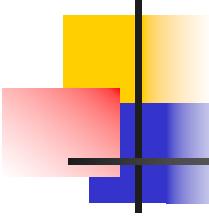
RIP2 协议的报文格式





RIP 协议的优缺点

- RIP 存在的一个问题是当网络出现故障时，要经过比较长的时间才能将此信息传送到所有的路由器。
- RIP 协议最大的优点就是实现简单，开销较小。
- RIP 限制了网络的规模，它能使用的最大距离为 15（16 表示不可达）。
- 路由器之间交换的路由信息是路由器中的完整路由表，因而随着网络规模的扩大，开销也就增加。



LS和DV的比较

消息复杂度

- LS: n 个节点, e 条链路,
 $O(ne)$ 条消息
- DV: 消息仅在邻居间传送

收敛速度

- LS: $O(n^2)$
 - 可能存在摆动
- DV: 依具体情况而定
 - 可能存在路由回路
 - 坏消息传播慢

鲁棒性: 某个路由器故障

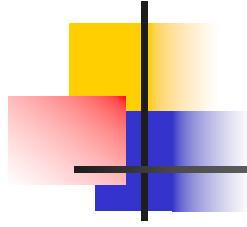
(或错误估计代价), 会发生什么情况?

LS:

- 结点广播链路失效
- 每个结点重新计算自己的路由表

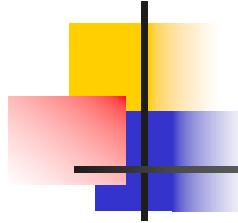
DV:

- 每个结点的路由信息均被其他结点使用
- 错误在整个网络中传播



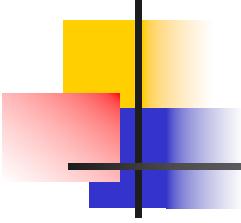
分层次的路由选择协议

- 因特网采用分层次的路由选择协议。
- 因特网的规模非常大。如果让所有的路由器知道所有的网络应怎样到达，则这种路由表将非常大，处理起来也太花时间。而所有这些路由器之间交换路由信息所需的带宽就会使因特网的通信链路饱和。
- 许多单位不愿意外界了解自己单位网络的布局细节和本部门所采用的路由选择协议（这属于本部门内部的事情），但同时还希望连接到因特网上。



自治系统 AS (Autonomous System)

- 自治系统 AS：
 - 在单一的技术管理下的一组路由器，而这些路由器使用一种 AS 内部的路由选择协议和共同的度量以确定分组在该 AS 内的路由。
 - 使用一种 AS 之间的路由选择协议用以确定分组在 AS 之间的路由。
- 强调：
 - 尽管一个 AS 使用了多种内部路由选择协议和度量，但重要的是一个 AS 对其他 AS 表现出来的是一个单一的和一致的路由选择策略。



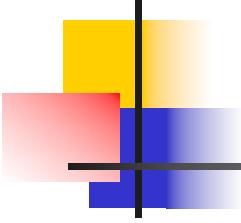
因特网有两大类路由选择协议

- **内部网关协议 IGP (Interior Gateway Protocol)**
即在一个自治系统内部使用的路由选择协议。目前这类路由选择协议使用得最多，如 RIP 和 OSPF 协议。
- **外部网关协议 EGP (External Gateway Protocol)**
若源站和目的站处在不同的自治系统中，当数据报传到一个自治系统的边界时，就需要使用一种协议将路由选择信息传递到另一个自治系统中。这样的协议就是外部网关协议 EGP。在外部网关协议中目前使用最多的是 BGP-4。

自治系统和 内部网关协议、外部网关协议

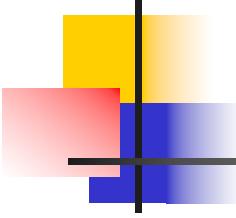


自治系统之间的路由选择也叫做
域间路由选择(interdomain routing),
在自治系统内部的路由选择叫做
域内路由选择(intradomain routing)



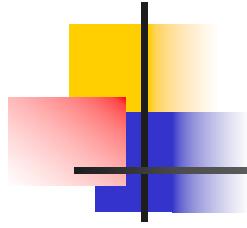
4.5.4 外部网关协议 BGP

- BGP 是不同自治系统的路由器之间交换路由信息的协议。
- BGP 较新版本是 2006 年 1 月发表的 BGP-4（BGP 第 4 个版本），即 RFC 4271 ~ 4278。
- 可以将 BGP-4 简写为 BGP。



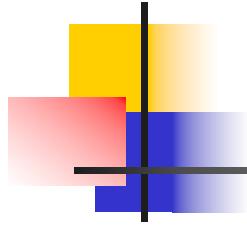
BGP 使用的环境却不同

- 因特网的规模太大，使得自治系统之间路由选择非常困难。对于自治系统之间的路由选择，要寻找最佳路由是很不现实的。
 - 当一条路径通过几个不同 AS 时，要想对这样的路径计算出有意义的代价是不太可能的。
 - 比较合理的做法是在 AS 之间交换“可达性”信息。
- 自治系统之间的路由选择必须考虑有关策略。
- 因此，边界网关协议 BGP 只能是力求寻找一条能够到达目的网络且**比较好的路由**（不能兜圈子），而**并非要寻找一条最佳路由**。



BGP 发言人(BGP speaker)

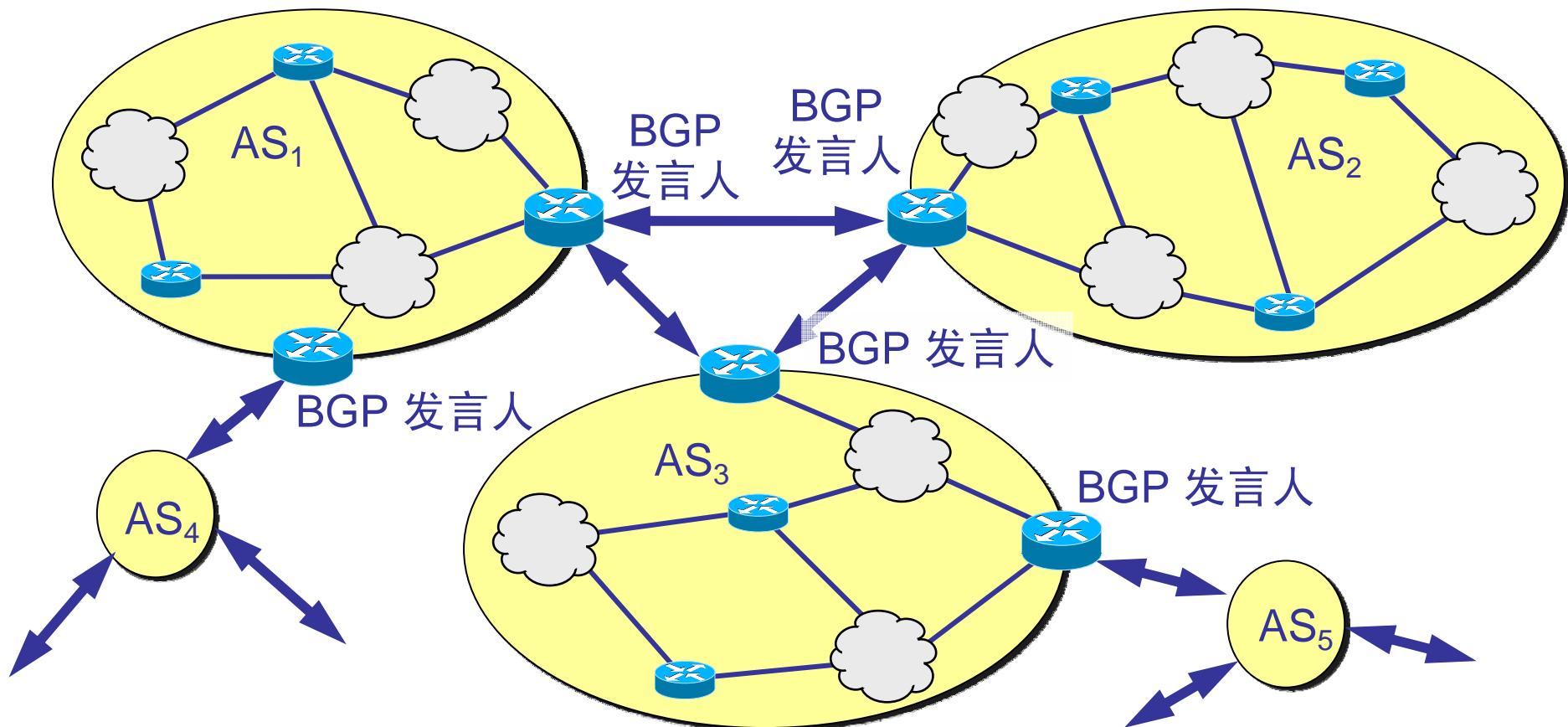
- 每一个自治系统的管理员要选择至少一个路由器作为该自治系统的“**BGP 发言人**”。
- 一般说来，两个 BGP 发言人都是通过一个共享网络连接在一起的，而 BGP 发言人往往就是 BGP 边界路由器，但也可以不是 BGP 边界路由器。

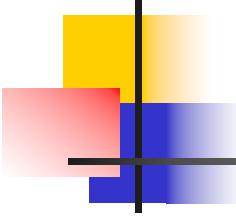


BGP 交换路由信息

- 一个 BGP 发言人与其他自治系统中的 BGP 发言人要交换路由信息，就要先建立 TCP 连接，然后在此连接上交换 BGP 报文以建立 BGP **会话**(session)，利用 BGP 会话交换路由信息。
- 使用 TCP 连接能提供可靠的服务，也简化了路由选择协议。
- 使用 TCP 连接交换路由信息的两个 BGP 发言人，彼此成为对方的邻站或对等站。

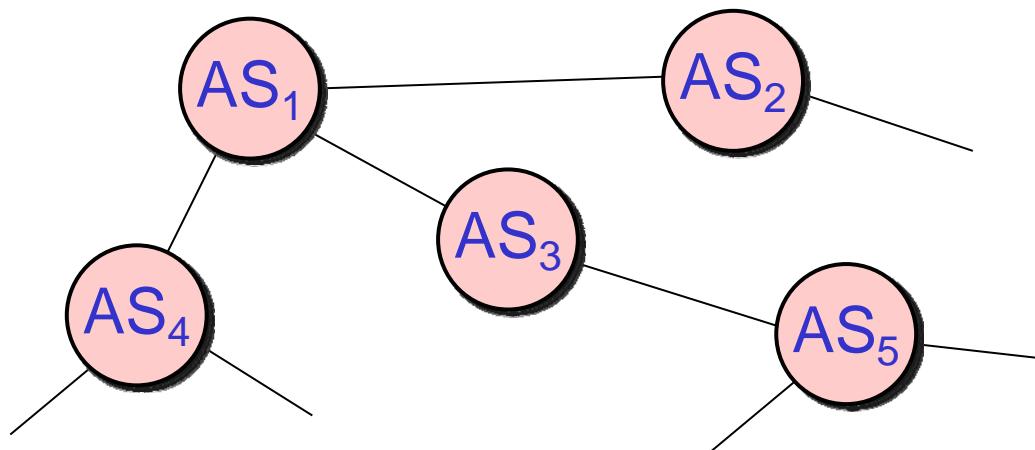
BGP 发言人和 自治系统 AS 的关系





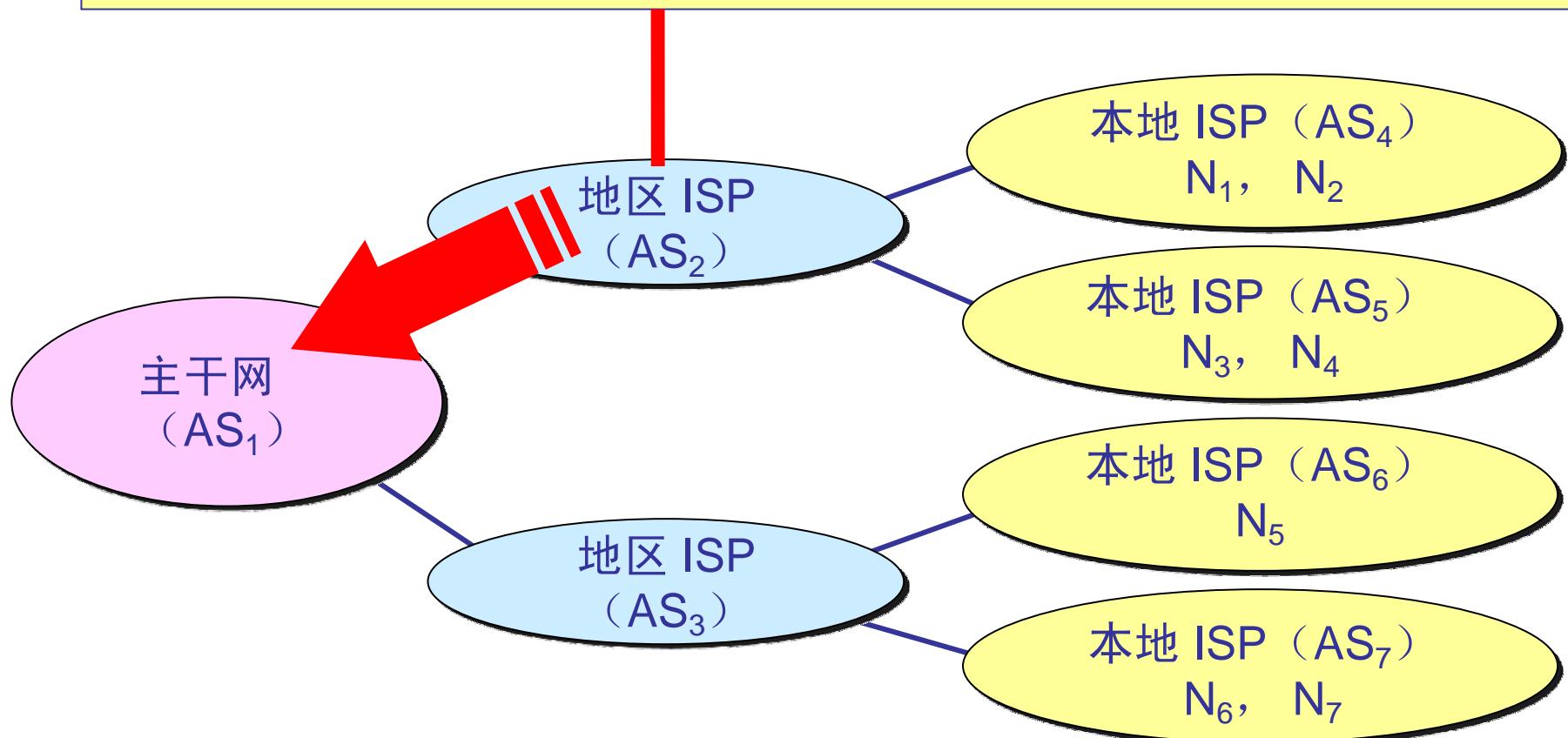
AS 的连通图举例

- BGP 所交换的网络可达性的信息就是要到达某个网络所要经过的一系列 AS。
- 当 BGP 发言人互相交换了网络可达性的信息后，各 BGP 发言人就根据所采用的策略从收到的路由信息中找出到达各 AS 的较好路由。



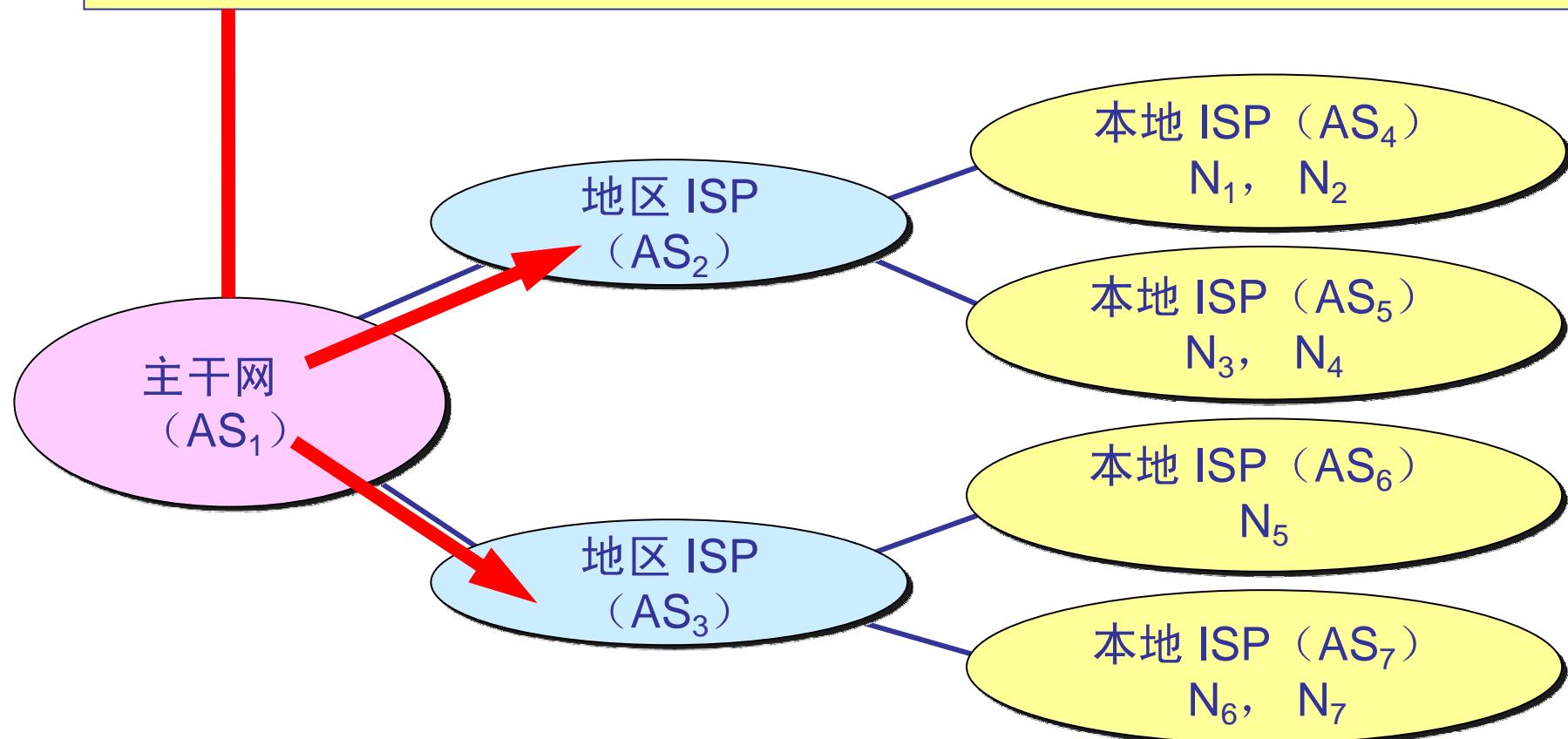
BGP 发言人交换路径向量

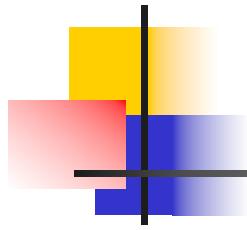
自治系统 AS_2 的 BGP 发言人通知主干网的 BGP 发言人：“要到达网络 N_1, N_2, N_3 和 N_4 可经过 AS_2 。”



BGP 发言人交换路径向量

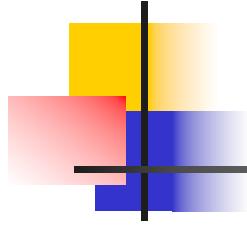
主干网还可发出通知：“要到达网络 N_5, N_6 和 N_7 可沿路径 (AS_1, AS_3) 。”





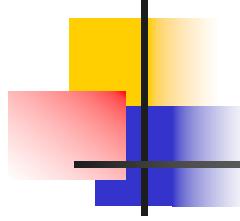
BGP 协议的特点

- BGP 协议交换路由信息的结点数量级是**自治系统数的量级**，这要比这些自治系统中的网络数少很多。
- 每一个自治系统中 BGP 发言人（或边界路由器）的数目是很少的。这样就使得自治系统之间的路由选择不致过分复杂。



BGP 协议的特点

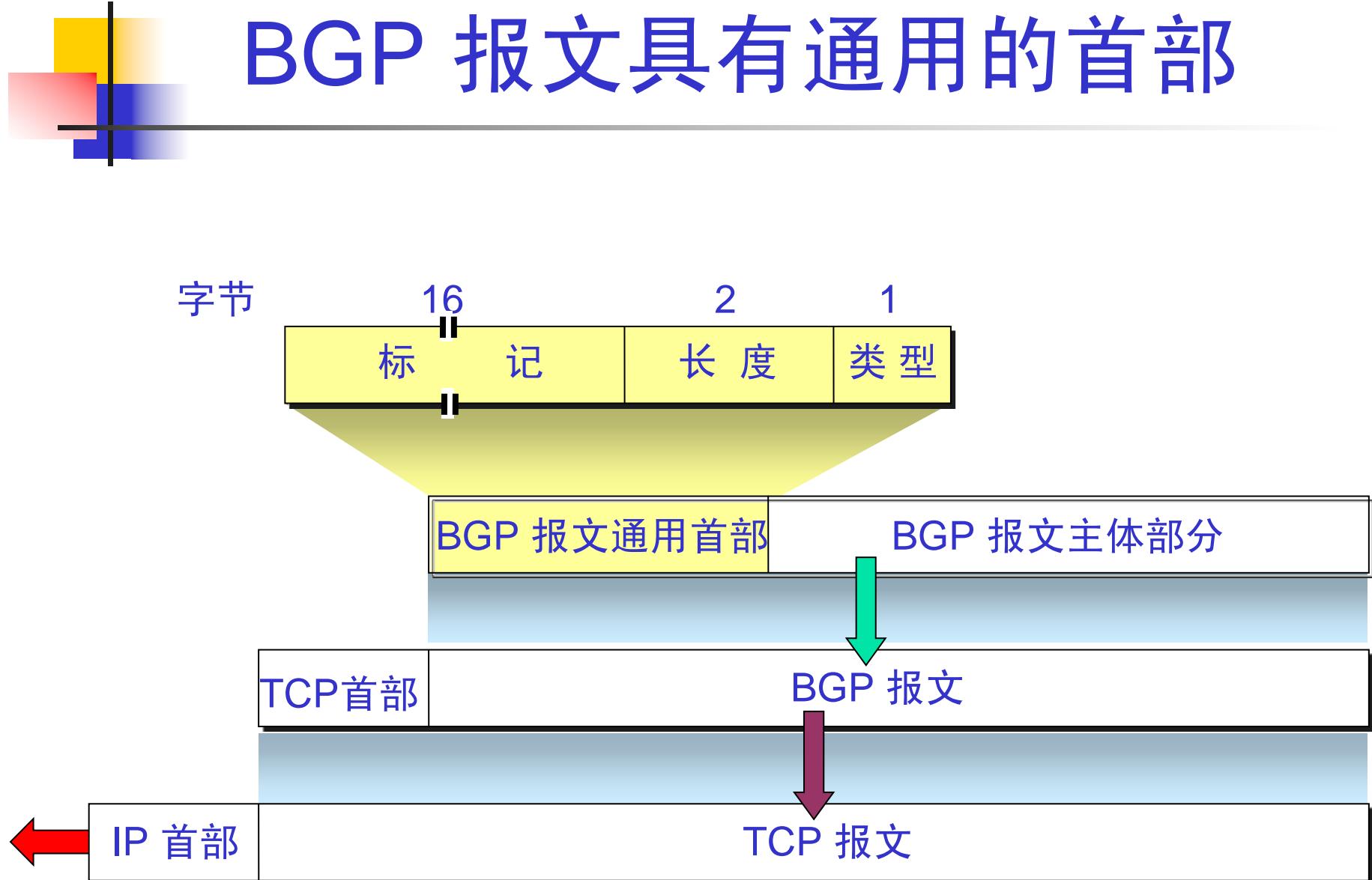
- BGP 支持 CIDR，因此 BGP 的路由表也就应当包括目的网络前缀、下一跳路由器，以及到达该目的网络所要经过的各个自治系统序列。
- 在BGP 刚刚运行时，BGP 的邻站是交换整个的 BGP 路由表。但以后只需要在发生变化时更新有变化的部分。这样做对节省网络带宽和减少路由器的处理开销方面都有好处。

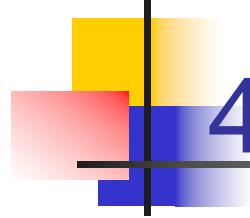


BGP-4 共使用四种报文

- (1) 打开(**OPEN**)报文，用来与相邻的另一个BGP发言人建立关系。
- (2) 更新(**UPDATE**)报文，用来发送某一路由的信息，以及列出要撤消的多条路由。
- (3) 保活(**KEEPALIVE**)报文，用来确认打开报文和周期性地证实邻站关系。
- (4) 通知(**NOTIFICATION**)报文，用来发送检测到的差错。
 - 在 RFC 2918 中增加了 ROUTE-REFRESH 报文，用来请求对等端重新通告。

BGP 报文具有通用的首部



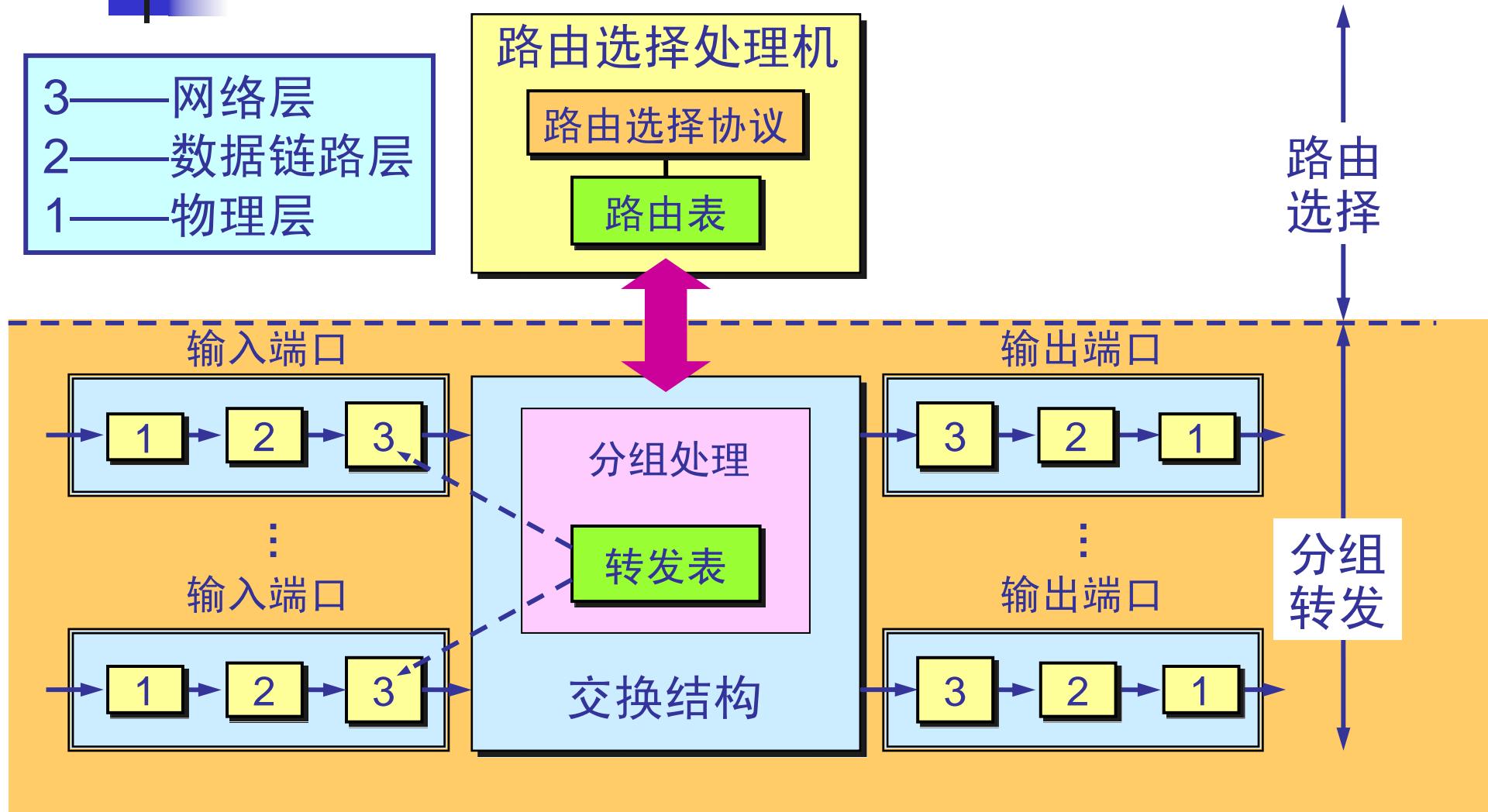


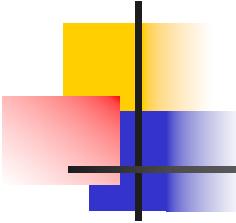
4.5.6 路由器在网际互连中的作用

1. 路由器的结构

- 路由器是一种具有多个输入端口和多个输出端口的专用计算机，其任务是转发分组。
- 路由器将某个输入端口收到的分组，按照分组要去的目的地（即目的网络），把该分组从路由器的某个合适的输出端口转发给下一跳路由器。
- 下一跳路由器也按照这种方法处理分组，直到该分组到达终点为止。

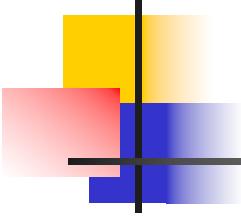
典型的路由器的结构





“转发”和“路由选择”的区别

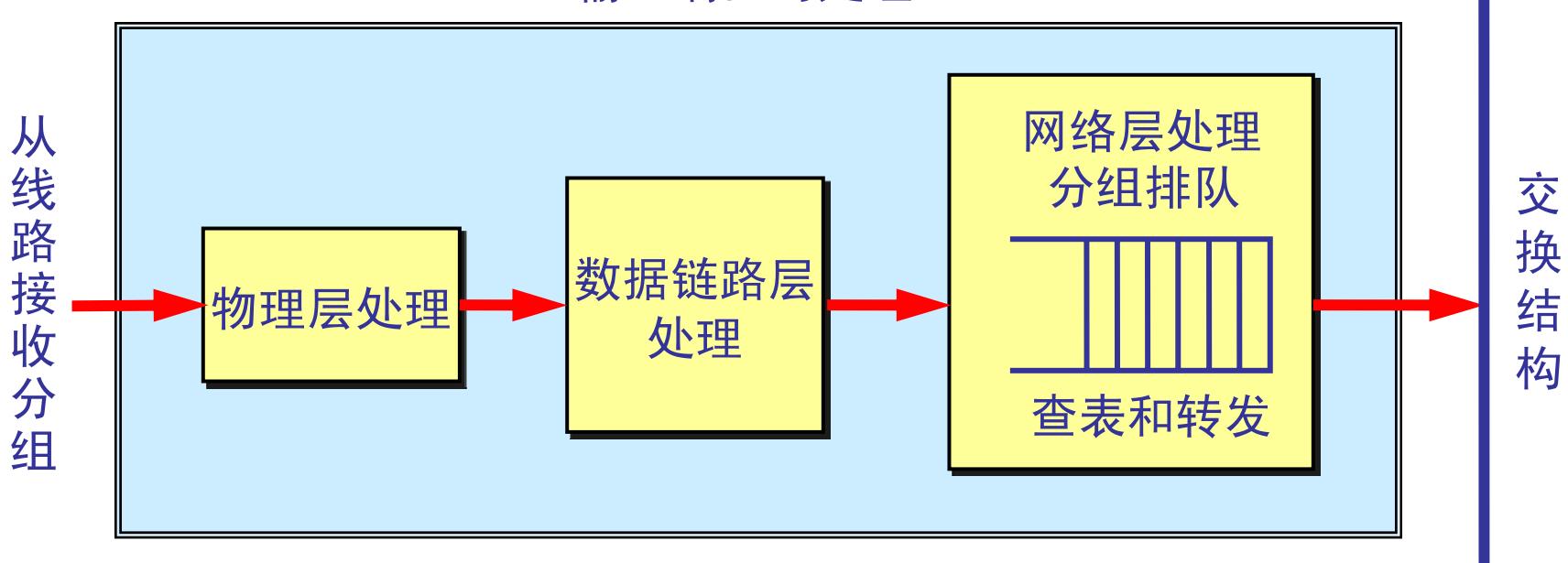
- “**转发**”(forwarding)就是路由器根据转发表将用户的 IP 数据报从合适的端口转发出去。
- “**路由选择**”(routing)则是按照分布式算法，根据从各相邻路由器得到的关于网络拓扑的变化情况，动态地改变所选择的路由。
- 路由表是根据路由选择算法得出的。而转发表是从路由表得出的。
- 在讨论路由选择的原理时，往往不去区分转发表和路由表的区别，



输入端口对线路上 收到的分组的处理

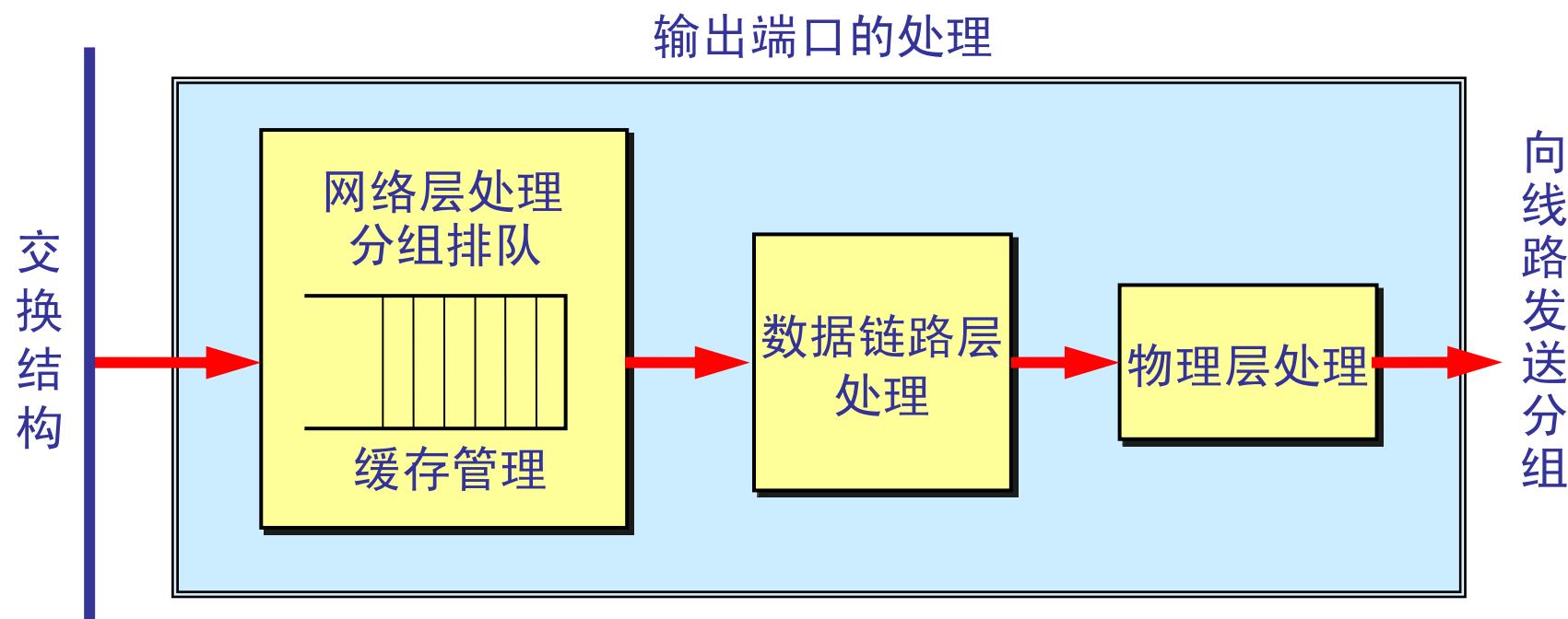
- 数据链路层剥去帧头部和尾部后，将分组送到网络层的队列中排队等待处理。这会产生一定的时延。

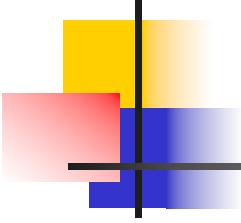
输入端口的处理



输出端口将交换结构传送 来的分组发送到线路

- 当交换结构传送过来的分组先进行缓存。数据链路层处理模块将分组加上链路层的首部和尾部，交给物理层后发送到外部线路。

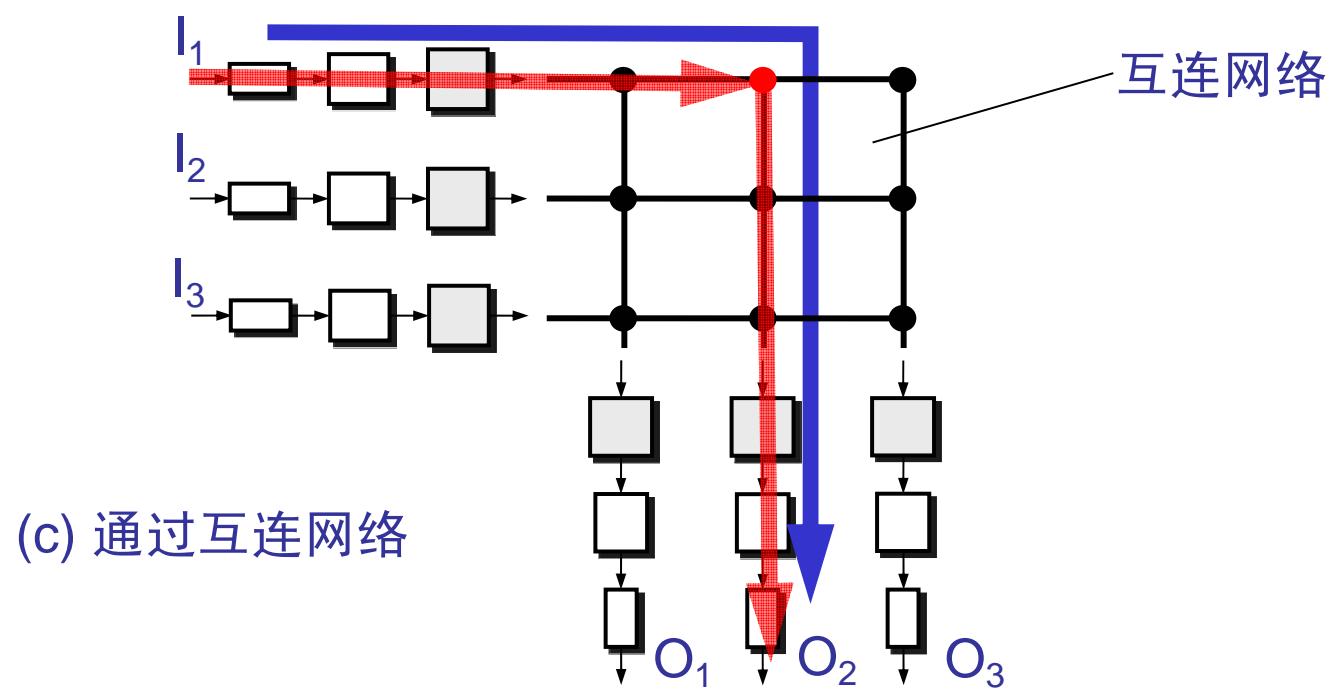
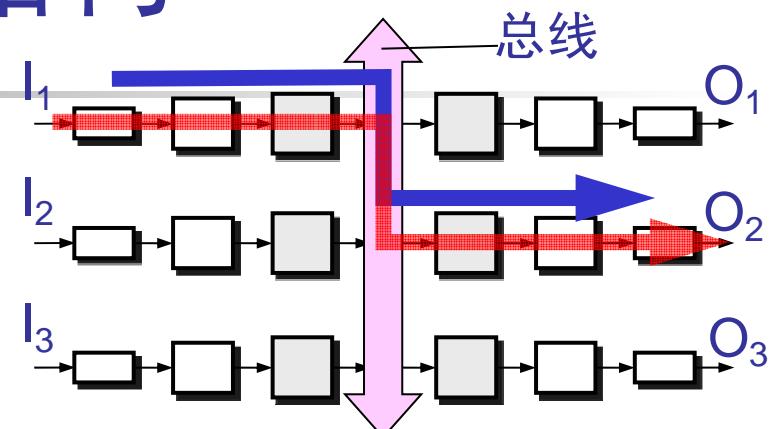
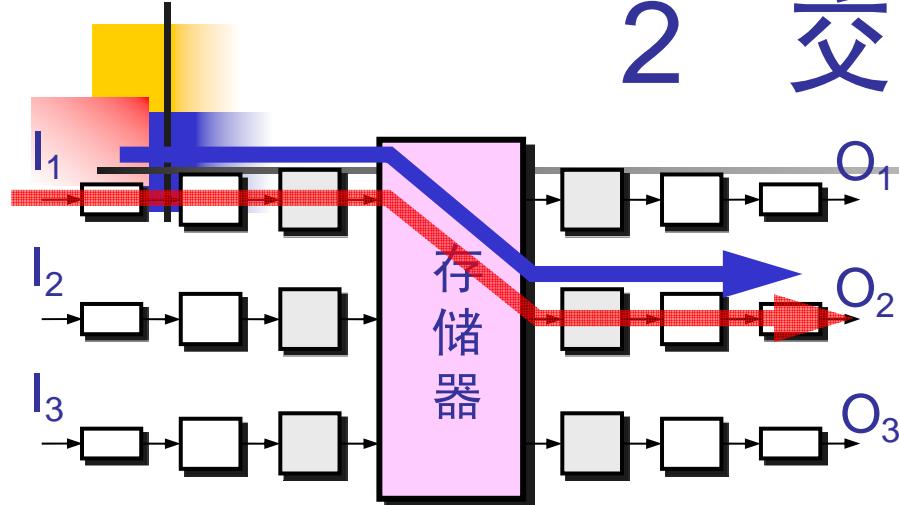




分组丢弃

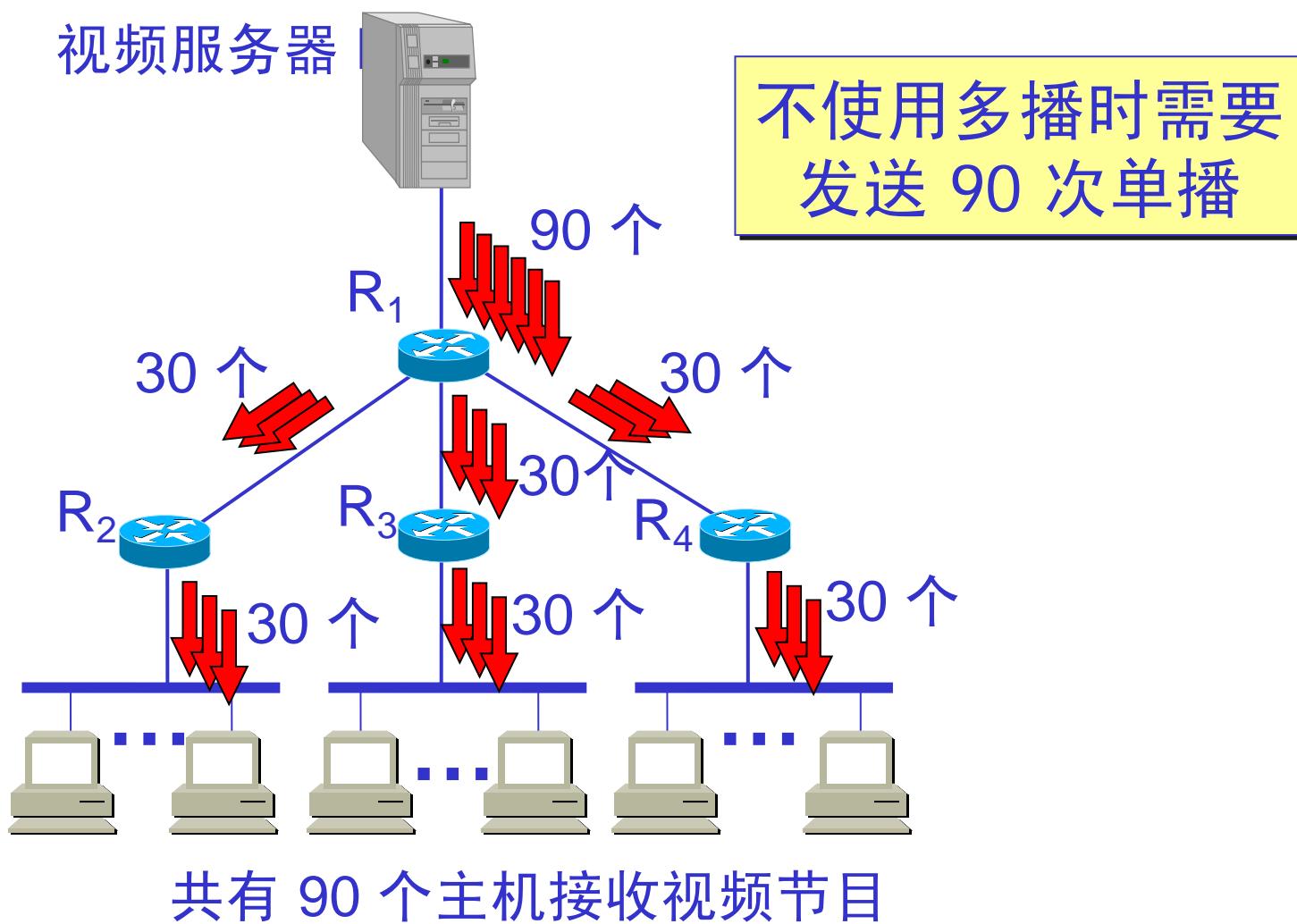
- 若路由器处理分组的速率赶不上分组进入队列的速率，则队列的存储空间最终必定减少到零，这就使后面再进入队列的分组由于没有存储空间而只能被丢弃。
- 路由器中的输入或输出队列产生溢出是造成分组丢失的重要原因。

2 交换结构

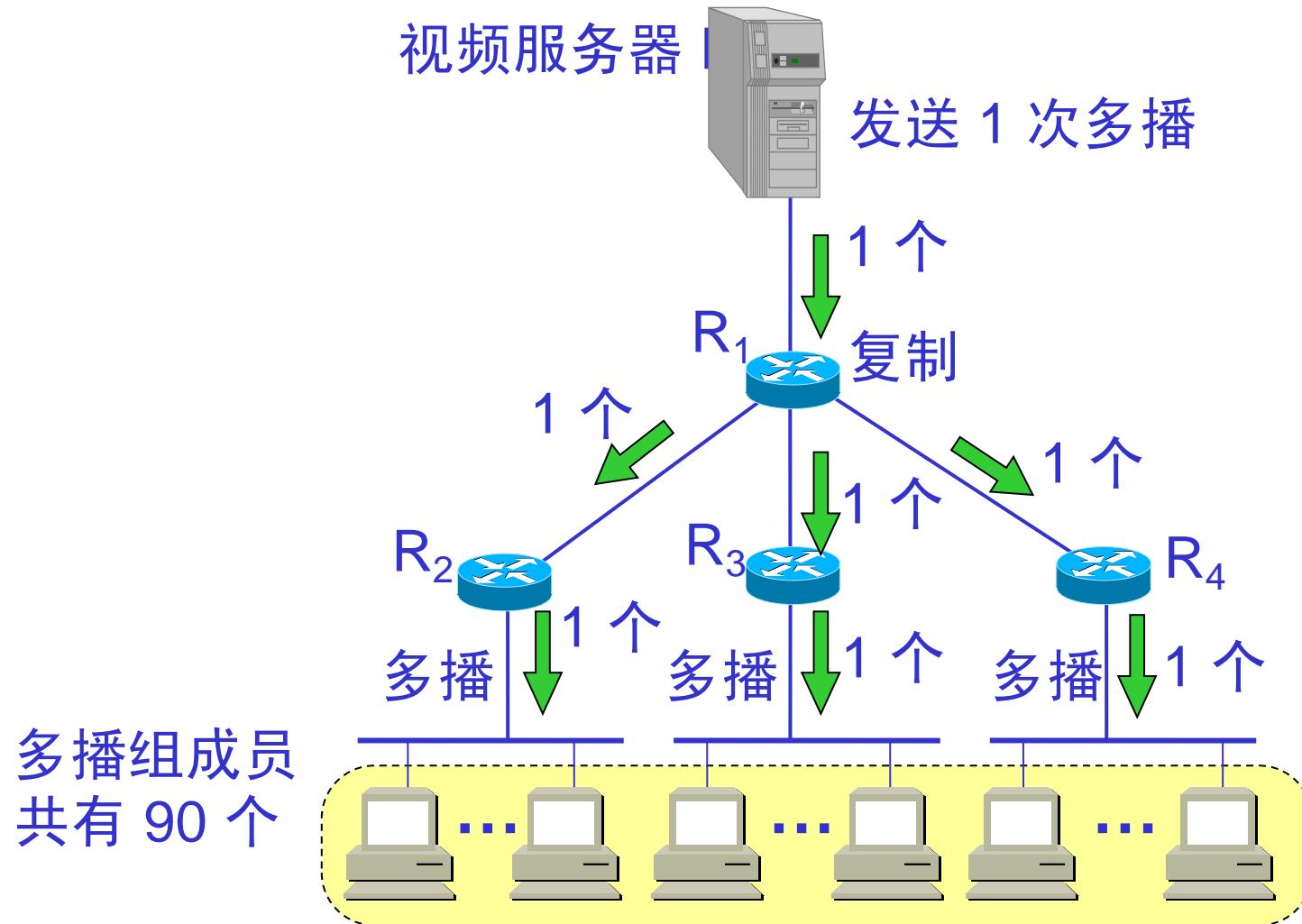


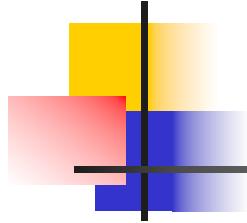
4.6 IP 多播

4.6.1 IP 多播的基本概念



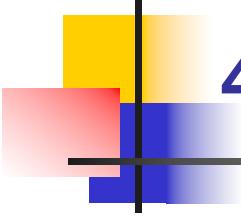
多播可明显地减少网络中资源的消耗





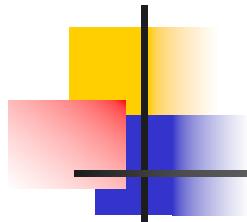
IP 多播的一些特点

- (1) 多播使用组地址——IP 使用 D 类地址支持多播。多播地址只能用于目的地址，而不能用于源地址。
- (2) 永久组地址——由因特网号码指派管理局 IANA 负责指派。
- (3) 动态的组成员
- (4) 使用硬件进行多播

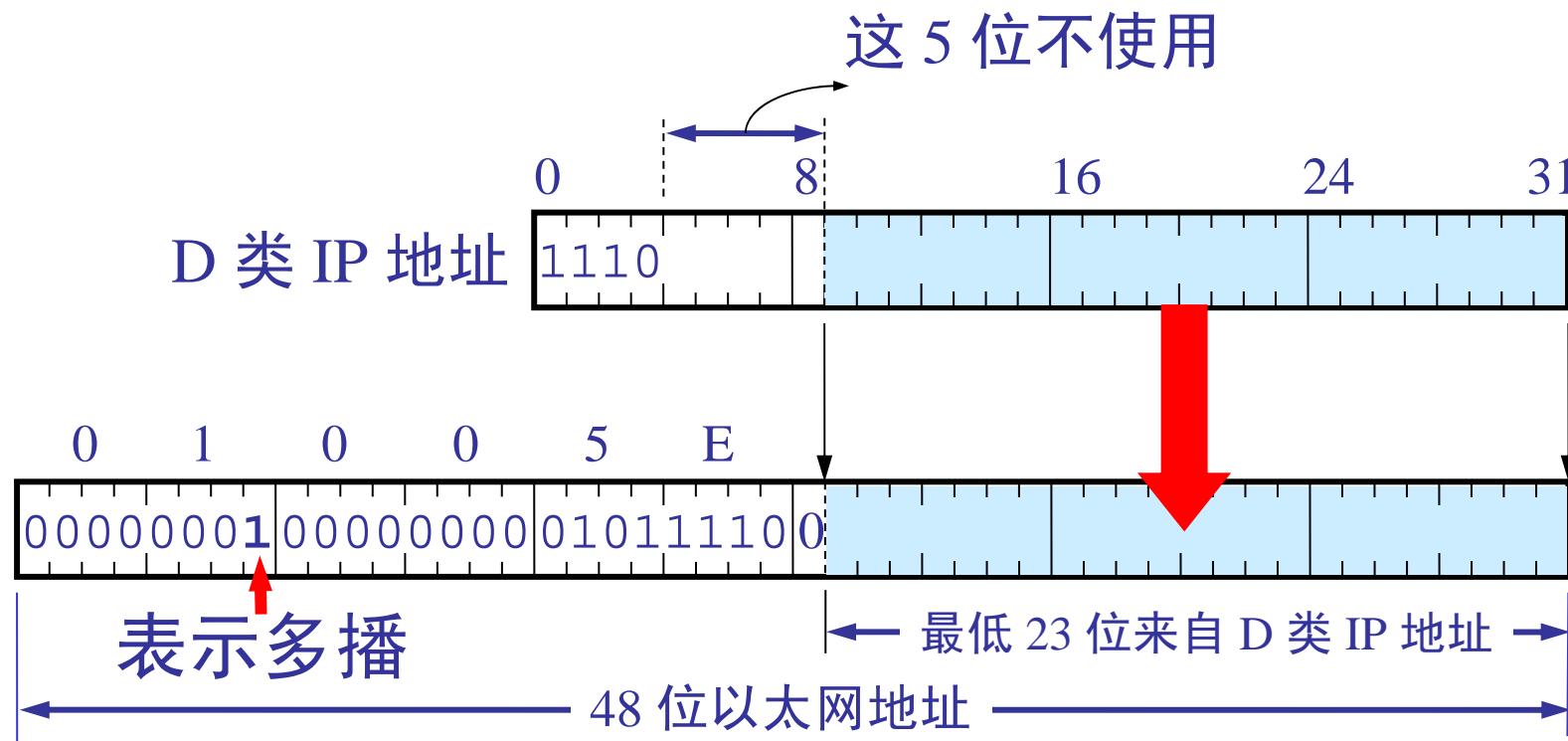


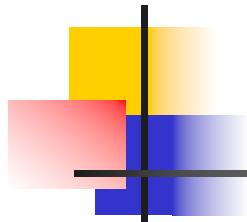
4.6.2 在局域网上进行硬件多播

- 因特网号码指派管理局 IANA 拥有的以太网地址块的高 24 位为 00-00-5E。
- 因此 TCP/IP 协议使用的以太网多播地址块的范围是：从 00-00-5E-00-00-00 到 00-00-5E-FF-FF-FF
- D 类 IP 地址可供分配的有 28 位，在这 28 位中的前 5 位不能用来构成以太网硬件地址。



D 类 IP 地址 与以太网多播地址的映射关系



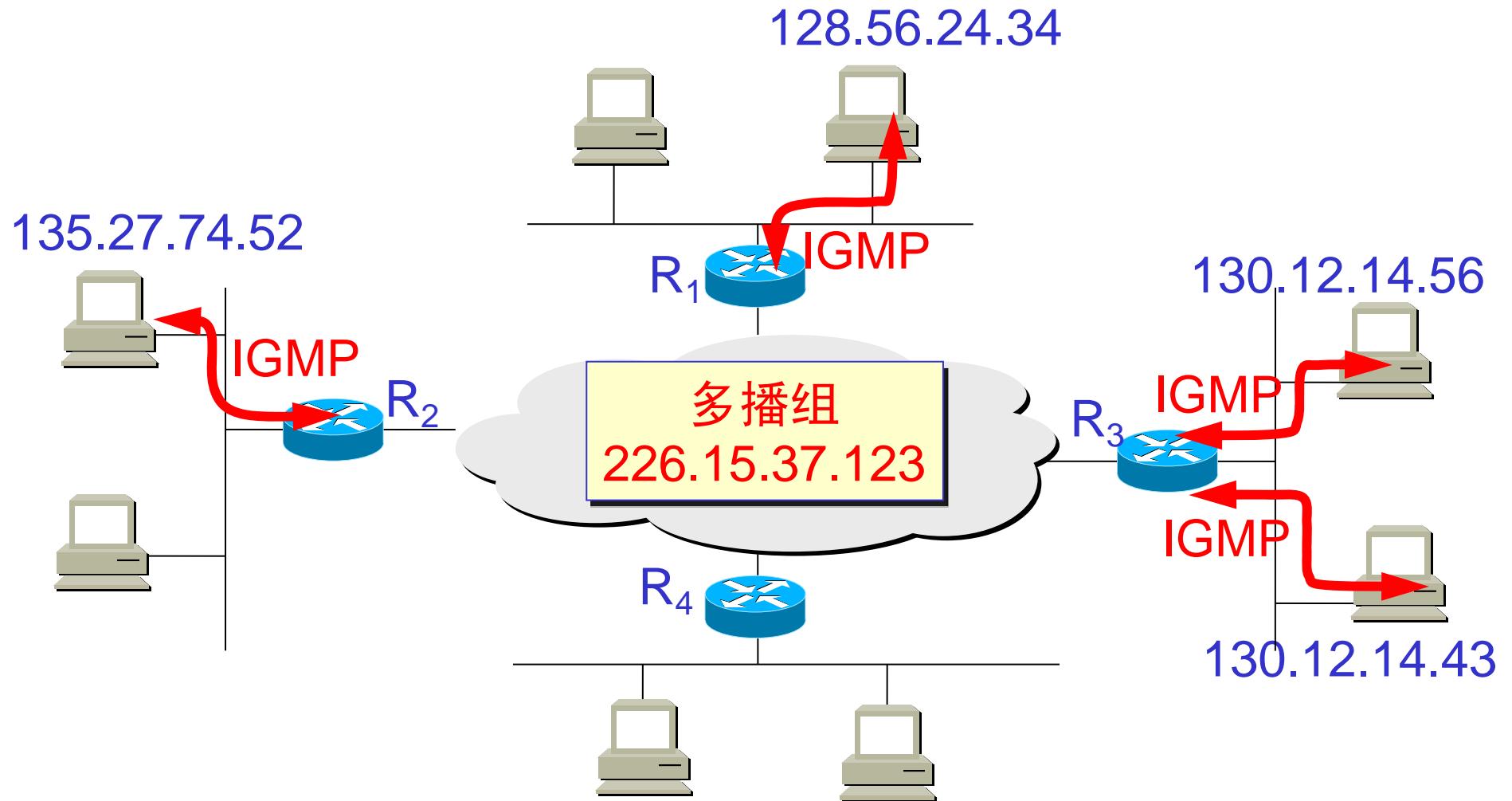


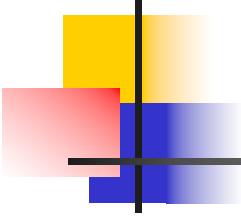
4.6.3 网际组管理协议 IGMP 和多播路由选择协议

1. IP多播需要两种协议

- 为了使路由器知道多播组成员的信息，需要利用**网际组管理协议 IGMP** (Internet Group Management Protocol)。
- 连接在局域网上的多播路由器还必须和因特网上的其他多播路由器协同工作，以便把多播数据报用最小代价传送给所有的组成员。这就需要使用**多播路由选择协议**。

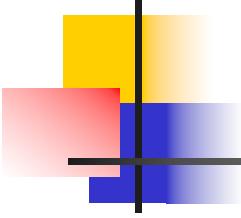
IGMP 使多播路由器知道多播组成员信息





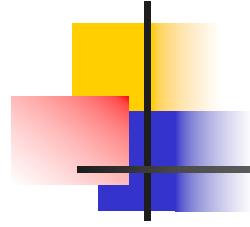
IGMP 的本地使用范围

- IGMP 并非在因特网范围内对所有多播组成员进行管理的协议。
- IGMP 不知道 IP 多播组包含的成员数，也不知道这些成员都分布在哪些网络上。
- IGMP 协议是让连接在本地局域网上的多播路由器知道本局域网上是否有主机（严格讲，是主机上的某个进程）参加了某个多播组。



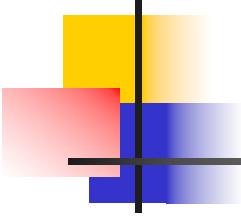
2. 网际组管理协议 IGMP

- 1989 年公布的 RFC 1112 (IGMPv1) 早已成为了因特网的标准协议。
- 1997 年公布的 RFC 2236 (IGMPv2, 建议标准) 对 IGMPv1 进行了更新。
- 2002 年 10 月公布了 RFC 3376 (IGMPv3, 建议标准) , 宣布 RFC 2236 (IGMPv2) 是陈旧的。



IGMP 是整个网际协议 IP 的一个组成部分

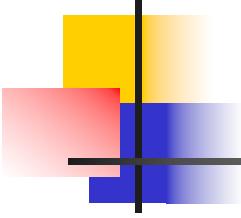
- 和 ICMP 相似，IGMP 使用 IP 数据报传递其报文（即 IGMP 报文加上 IP 首部构成 IP 数据报），但它也向 IP 提供服务。
- 我们不把 IGMP 看成是一个单独的协议，而是属于整个网际协议 IP 的一个组成部分。



IGMP 可分为两个阶段

■ 第一阶段：

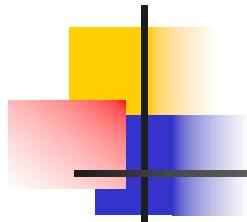
- 当某个主机加入新的多播组时，该主机应向多播组的多播地址发送IGMP 报文，声明自己要成为该组的成员。
- 本地的多播路由器收到 IGMP 报文后，将组成员关系转发给因特网上的其他多播路由器。



IGMP 可分为两个阶段

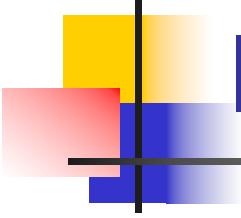
■ 第二阶段：

- 因为组成员关系是**动态的**，因此本地多播路由器要**周期性地探询**本地局域网上的主机，以便知道这些主机是否还继续是组的成员。
- 只要对某个组有一个主机响应，那么多播路由器就认为这个组是活跃的。
- 但一个组在经过几次的探询后仍然没有一个主机响应，则不再将该组的成员关系转发给其他的多播路由器。



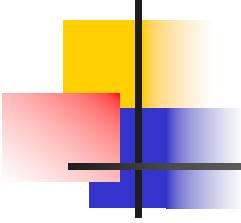
IGMP 采用的一些具体措施

- 在主机和多播路由器之间的所有通信都是使用 IP 多播。
- 多播路由器在探询组成员关系时，只需要对所有的组发送一个请求信息的询问报文，而不需要对每一个组发送一个询问报文。默认的询问速率是每 125 秒发送一次。
- 当同一个网络上连接有几个多播路由器时，它们能够迅速和有效地选择其中的一个来探询主机的成员关系。



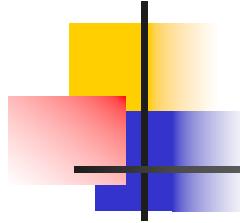
IGMP 采用的一些具体措施（续）

- 在 IGMP 的询问报文中有—个数值 N ，它指明一个最长响应时间（默认值为 10 秒）。当收到询问时，主机在 0 到 N 之间随机选择发送响应所需经过的时延。对应于最小时延的响应最先发送。
- 同一个组内的每一个主机都要监听响应，只要有本组的其他主机先发送了响应，自己就可以不再发送响应了。



3. 多播路由选择

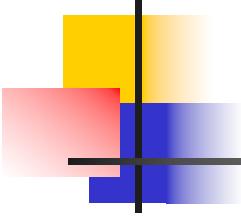
- 多播路由选择协议尚未标准化。
- 一个多播组中的成员是动态变化的，随时会有主机加入或离开这个多播组。
- 多播路由选择实际上就是要找出以源主机为根结点的多播转发树。
- 在多播转发树上的路由器不会收到重复的多播数据报。
- 对不同的多播组对应于不同的多播转发树。同一个多播组，对不同的源点也会有不同的多播转发树。



转发多播数据报使用的方法

(1) 洪泛与剪除

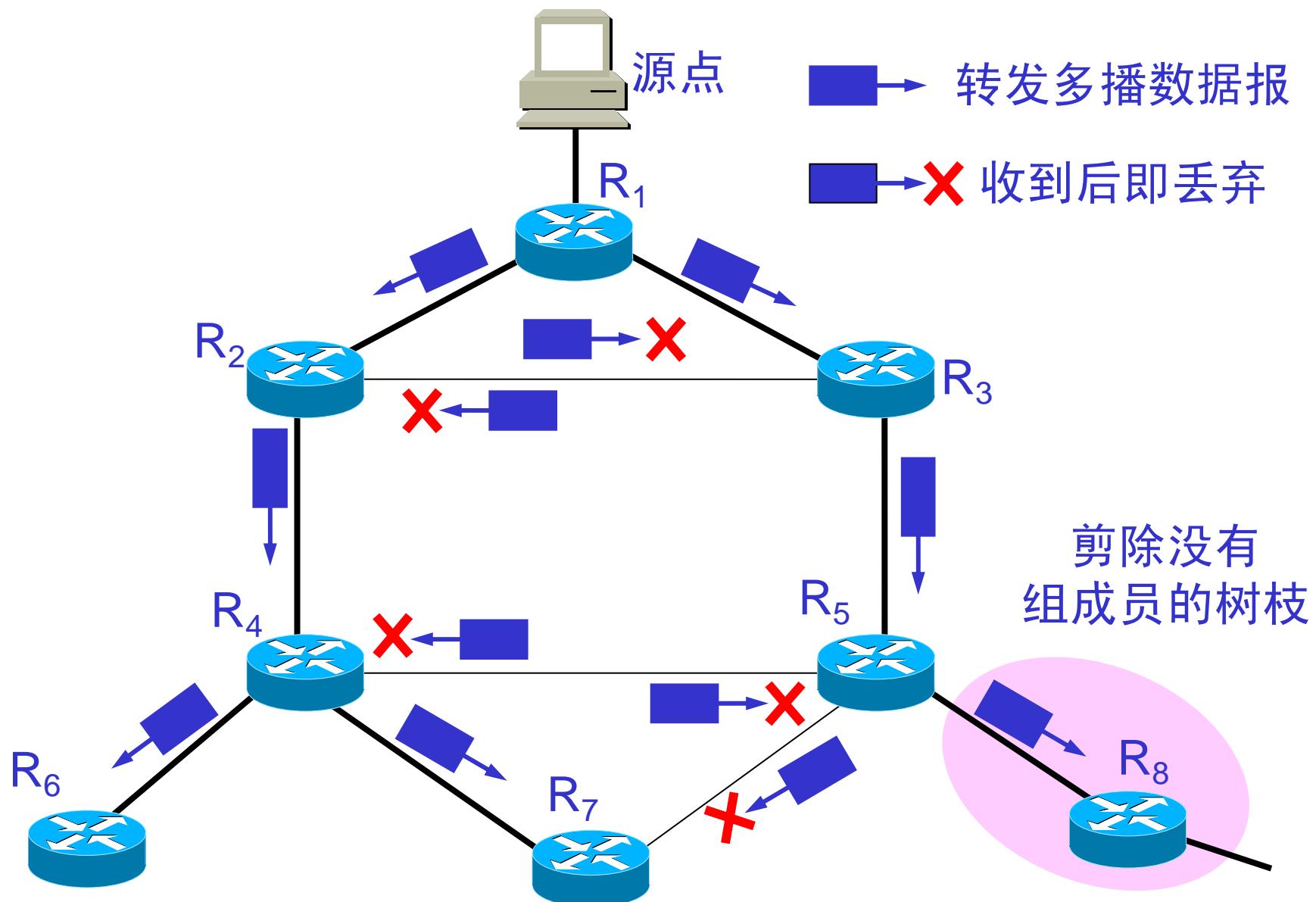
- 这种方法适合于较小的多播组，而所有的组成员接入的局域网也是相邻接的。
- 一开始，路由器转发多播数据报使用洪泛的方法（这就是广播）。为了避免兜圈子，采用了叫做**反向路径广播 RPB** (Reverse Path Broadcasting)的策略。



RPB 的要点

- 路由器收到多播数据报时，先检查是否从源点经最短路径传送来的。
- 若是，就向所有其他方向转发刚才收到的多播数据报（但进入的方向除外），否则就丢弃而不转发。
- 如果存在几条同样长度的最短路径），那么只能选择一条最短路径，选择的准则就是看这几条最短路径中的相邻路由器谁的IP地址最小。

反向路径广播 RPB 和剪除

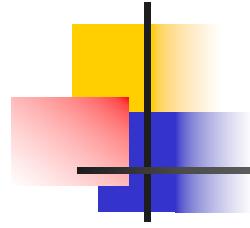


4.7

虚拟专用网 VPN 和网络地址转换 NAT

4.7.1 虚拟专用网 VPN

- **本地地址**——仅在机构内部使用的 IP 地址，可以由本机构自行分配，而不需要向因特网的管理机构申请。
- **全球地址**——全球唯一的IP地址，必须向因特网的管理机构申请。



RFC 1918 指明的专用地址

(private address)

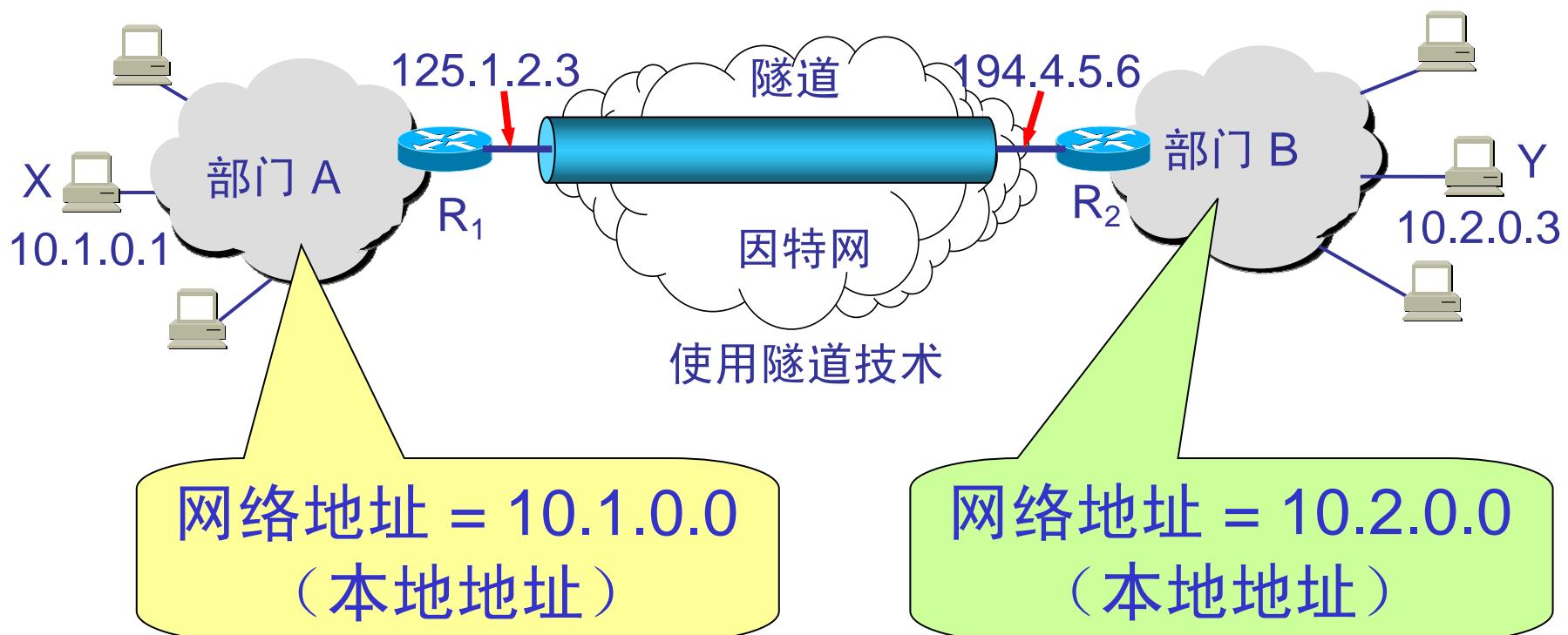
- 10.0.0.0 到 10.255.255.255
- 172.16.0.0 到 172.31.255.255
- 192.168.0.0 到 192.168.255.255
- 这些地址只能用于一个机构的内部通信，而不能用于和因特网上的主机通信。
- 专用地址只能用作本地地址而不能用作全球地址。在因特网中的所有路由器对目的地址是专用地址的数据报一律不进行转发。

用隧道技术实现虚拟专用网

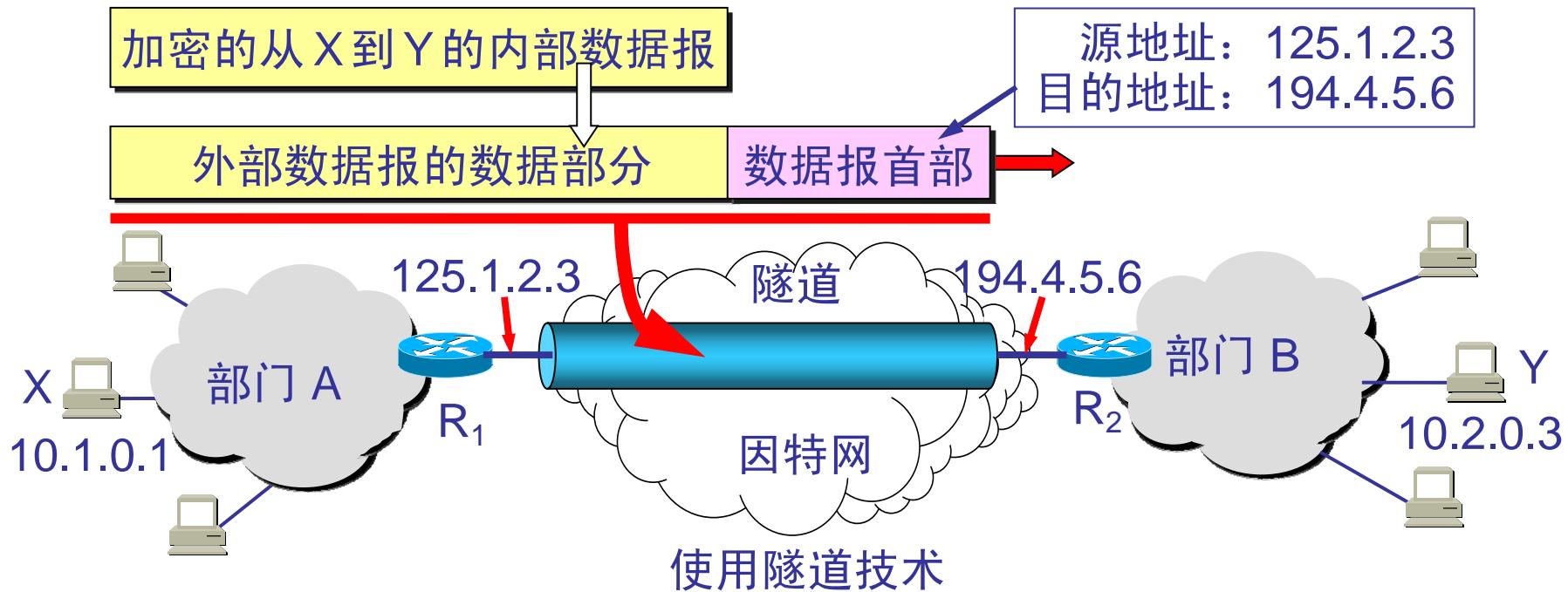
本地地址

全球地址

本地地址



用隧道技术实现虚拟专用网

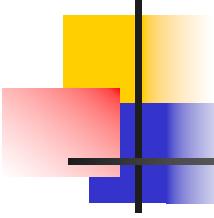


内联网 intranet 和外联网 extranet

(都是基于 TCP/IP 协议)

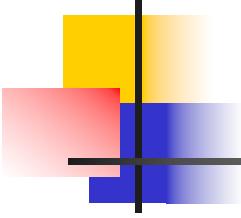
- 由部门 A 和 B 的内部网络所构成的虚拟专用网 VPN 又称为 **内联网(intranet)**，表示部门 A 和 B 都是在**同一个机构**的内部。
- 一个机构和某些**外部机构**共同建立的虚拟专用网 VPN 又称为**外联网(extranet)**。





远程接入VPN(remote access VPN)

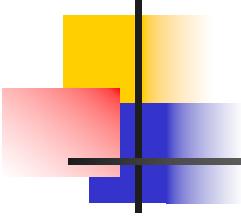
- 有的公司可能没有分布在不同场所的部门，但有很多流动员工在外地工作。公司需要和他们保持联系，远程接入 VPN 可满足这种需求。
- 在外地工作的员工拨号接入因特网，而驻留在员工 PC 机中的 VPN 软件可在员工的 PC 机和公司的主机之间建立 VPN 隧道，因而外地员工与公司通信的内容是保密的，员工们感到好像就是使用公司内部的本地网络。



4.7.2 网络地址转换 NAT

(Network Address Translation)

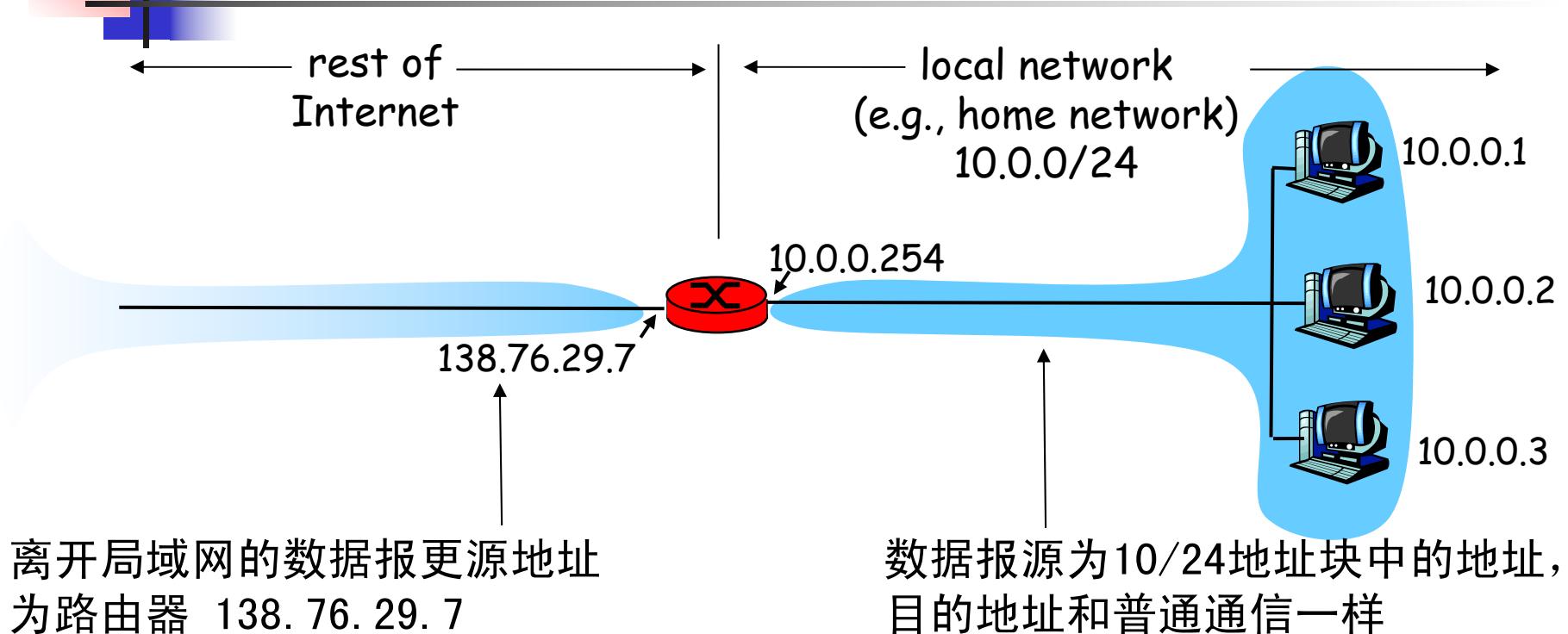
- 网络地址转换 NAT 方法于1994年提出。
- 需要在专用网连接到因特网的路由器上安装 NAT 软件。装有 NAT 软件的路由器叫做 NAT 路由器，它至少有一个有效的外部全球地址 IP_G 。
- 所有使用本地地址的主机在和外界通信时都要在 NAT 路由器上将其本地地址转换成 IP_G 才能和因特网连接。



网络地址转换的过程

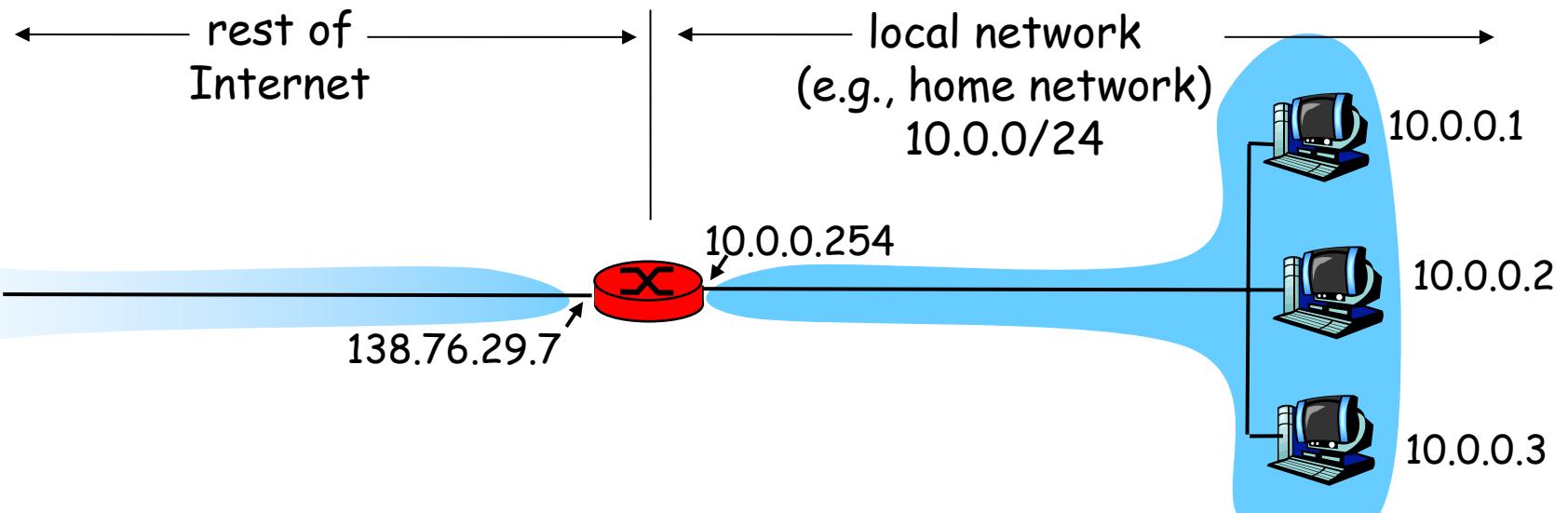
- 内部主机 X 用本地地址 IP_x 和因特网上主机 Y 通信所发送的数据报必须经过 NAT 路由器。
- NAT 路由器将数据报的源地址 IP_x 转换成全球地址 IP_G ，但目的地址 IP_Y 保持不变，然后发送到因特网。
- NAT 路由器收到主机 Y 发回的数据报时，知道数据报中的源地址是 IP_Y 而目的地址是 IP_G 。
- 根据 NAT 转换表，NAT 路由器将目的地址 IP_G 转换为 IP_x ，转发给最终的内部主机 X。

NAT: Network Address Translation



字段	旧IP地址	新IP地址	目的地址
源IP	10.0.0.1	138.76.29.7	222.1.1.12
源IP	10.0.0.2	138.76.29.7	211.6.4.8

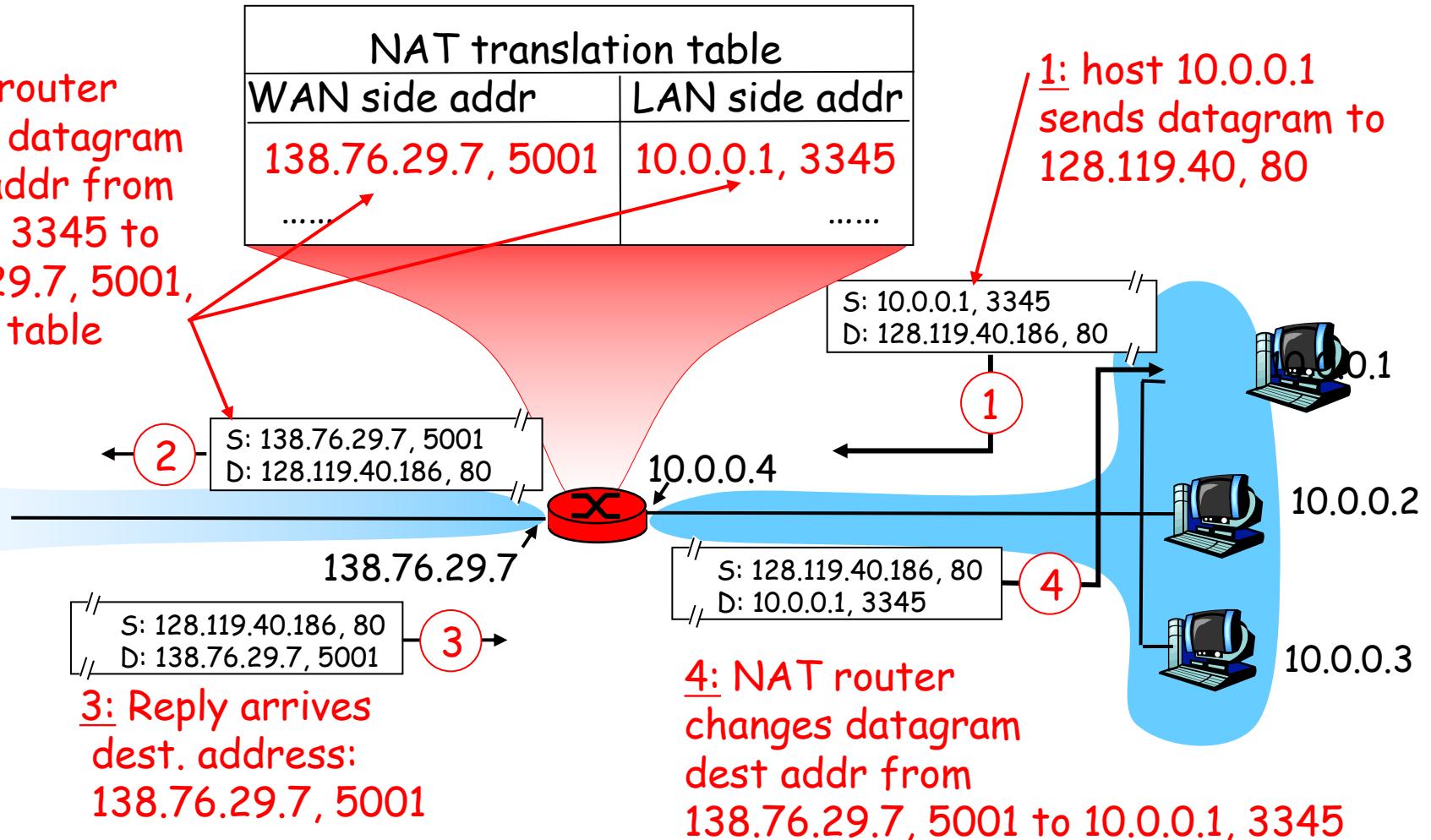
使用端口号的NAT

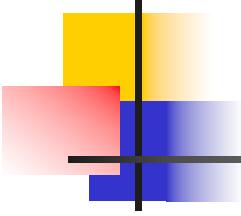


方向	字段	旧IP地址: 端口号	新IP地址: 端口号
出	源IP: 端口号	10.0.0.1: 3345	138.76.29.7: 5001
入	目的IP: 端口号	138.76.29.7: 5001	10.0.0.1: 3345
出	源IP: 端口号	10.0.0.2: 3345	138.76.29.7: 5002
入	目的IP: 端口号	138.76.29.7: 5002	10.0.0.2: 3345

NAT: Network Address Translation

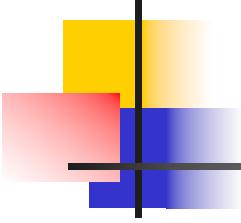
2: NAT router changes datagram source addr from 10.0.0.1, 3345 to 138.76.29.7, 5001, updates table





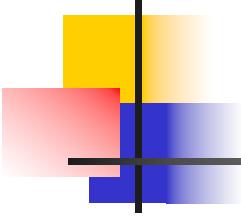
使用端口号的NAT实现

- 向外的报文
 - 更换(*source IP address, port #*) 为(*NAT IP address, new port #*)
 - 远端主机使用(*NAT IP address, new port #*) 进行应答
- 在本地转换表中记录
 - 所有的翻译对 (*source IP address, port #*) to (*NAT IP address, new port #*)
- 进入的数据报
 - 更换目的地址中的 (*NAT IP address, new port #*)为本地转换表中的(*source IP address, port #*)



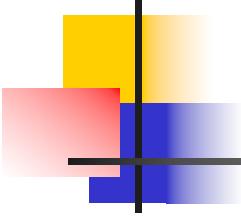
NAT特点

- 本地网络只要有一个公网IP就可以接入互联网：
 - 不需要向ISP申请连续的地址块，一个IP满足多个上网设备
 - 局域网内可以自由修改网络设备的地址而通知外部网络
 - 自由更换ISP，而不需修改内部网络地址
 - 局域网内部设备不需要被网络外主机寻址（即可见），增强了安全性
 - 16bit端口号可以支持同时60000个的链接
 - 工作在网络层速度快
- 破坏TCP/IP协议族端到端的可寻址的初衷——应使用IPV6来解决地址不足的问题。



6.8 下一代网际协议 IPv6

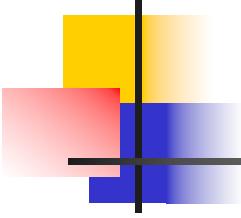
- 从计算机本身发展以及从因特网规模和网络传输速率来看，现在 IPv4 已很不适用。
- 最主要的问题就是 32 bit 的 IP 地址不够用。
- 要解决 IP 地址耗尽的问题的措施：
 - 采用无类别编址 CIDR，使 IP 地址的分配更加合理。
 - 采用网络地址转换 NAT 方法以节省全球 IP 地址。
 - 采用具有更大地址空间的新版本的 IP 协议 IPv6。



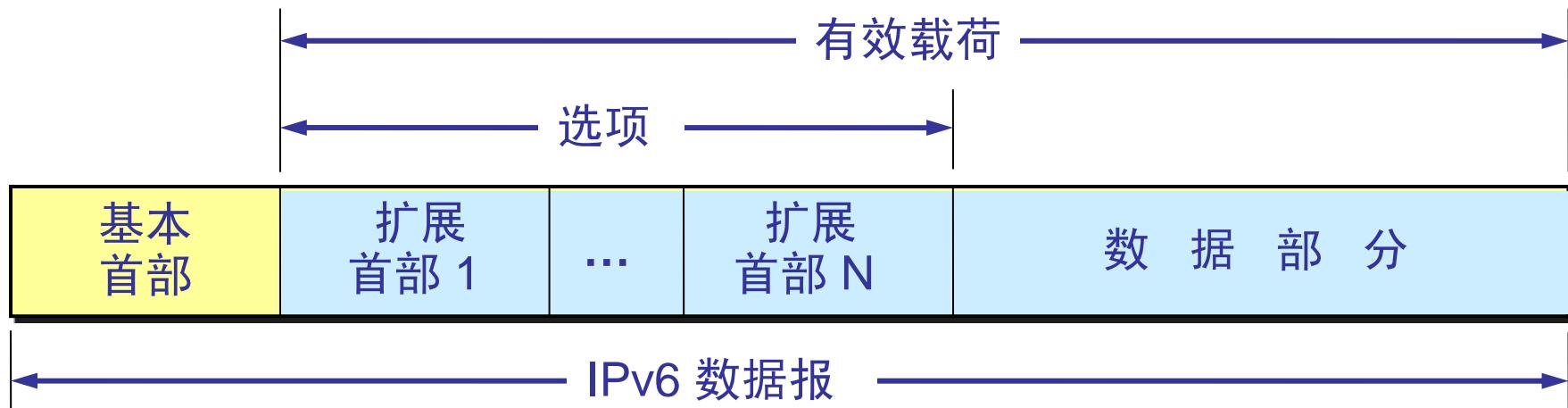
6.8.2 IPv6 的基本首部

■ IPv6 所引进的主要变化如下

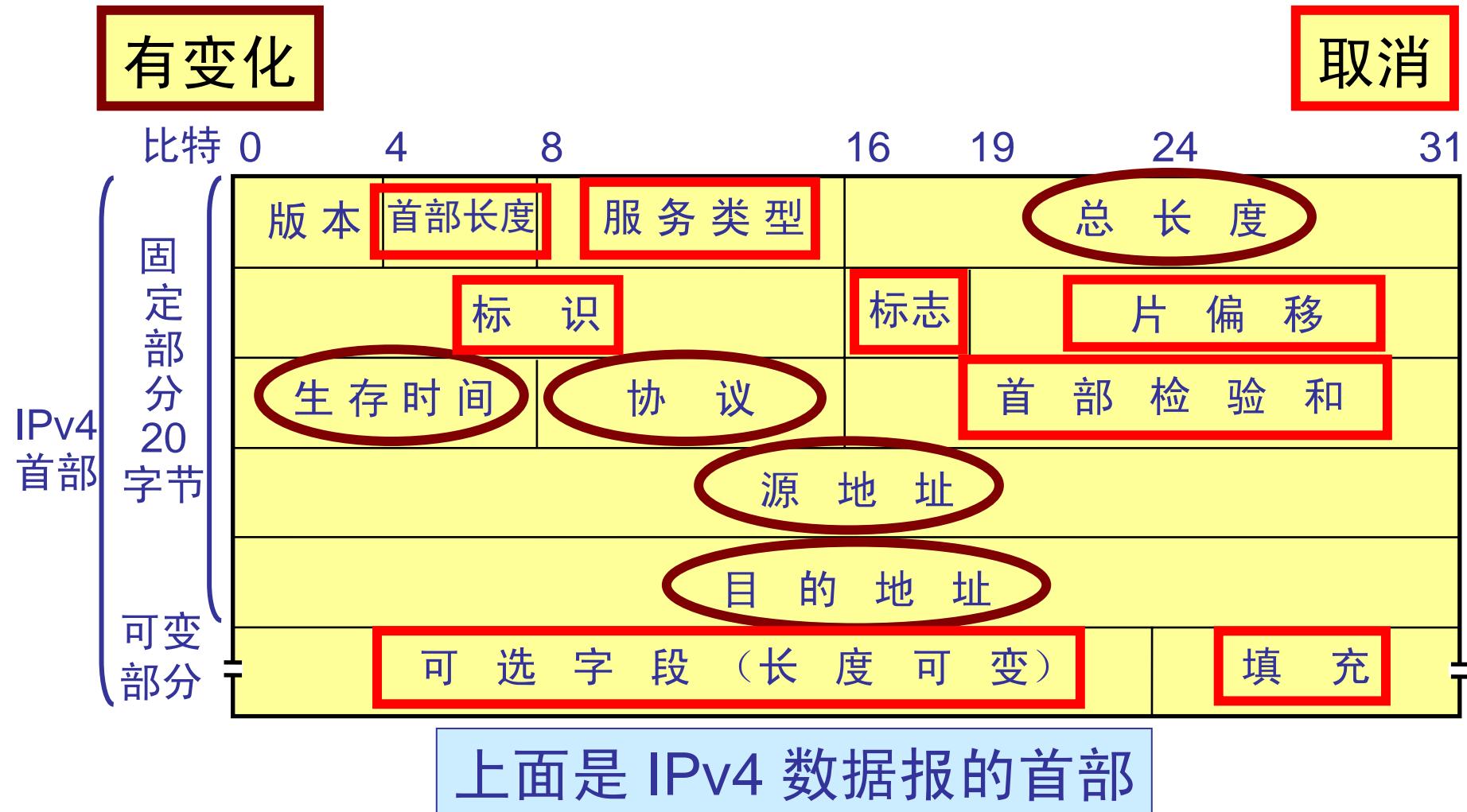
- 更大的地址空间：IPv6 将地址从 IPv4 的 32 bit 增大到了 128 bit
- 扩展的地址层次结构
- 灵活的头部格式
- 改进的选项
- 允许协议继续扩充
- 支持即插即用（即自动配置）
- 支持资源的预分配

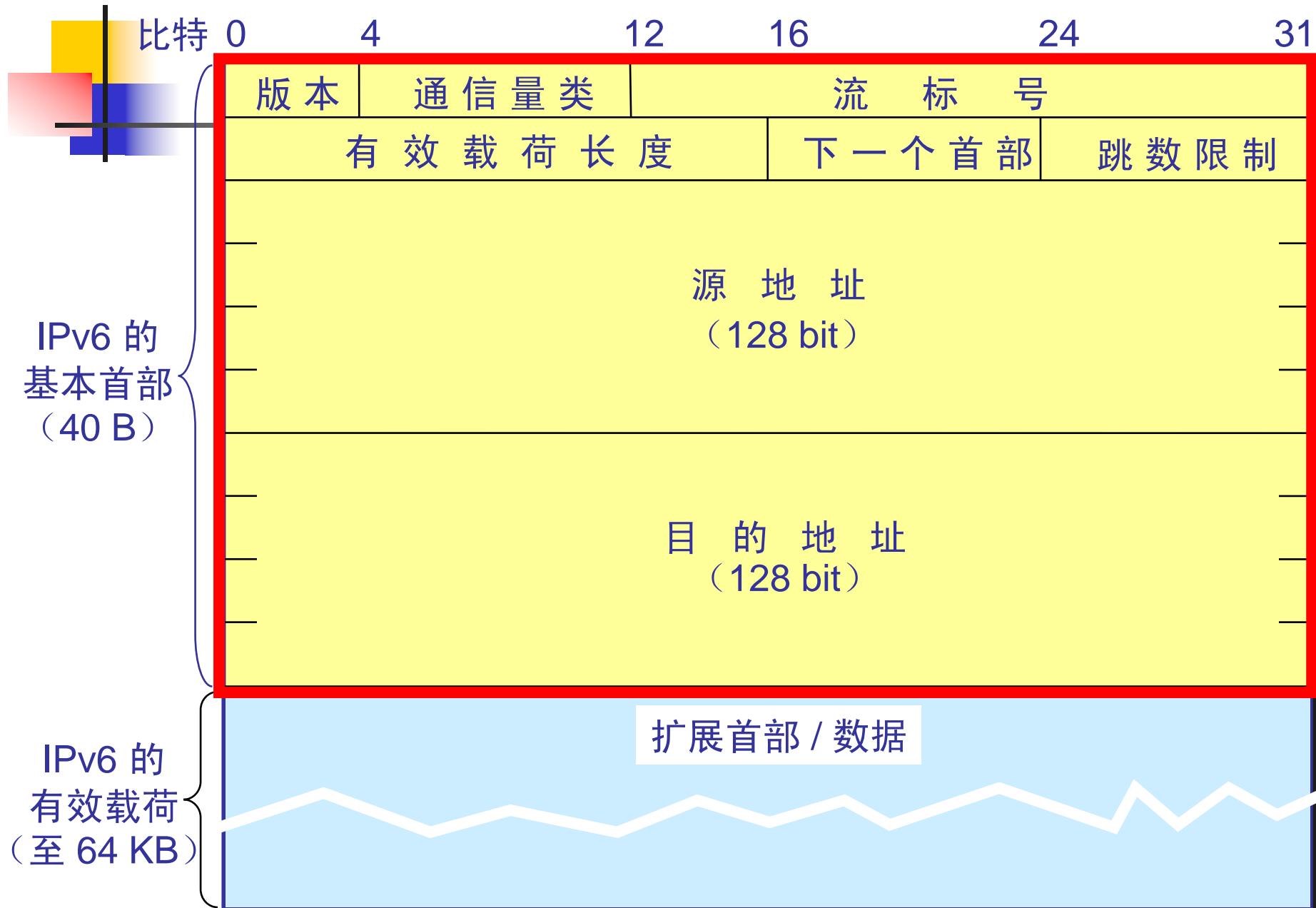


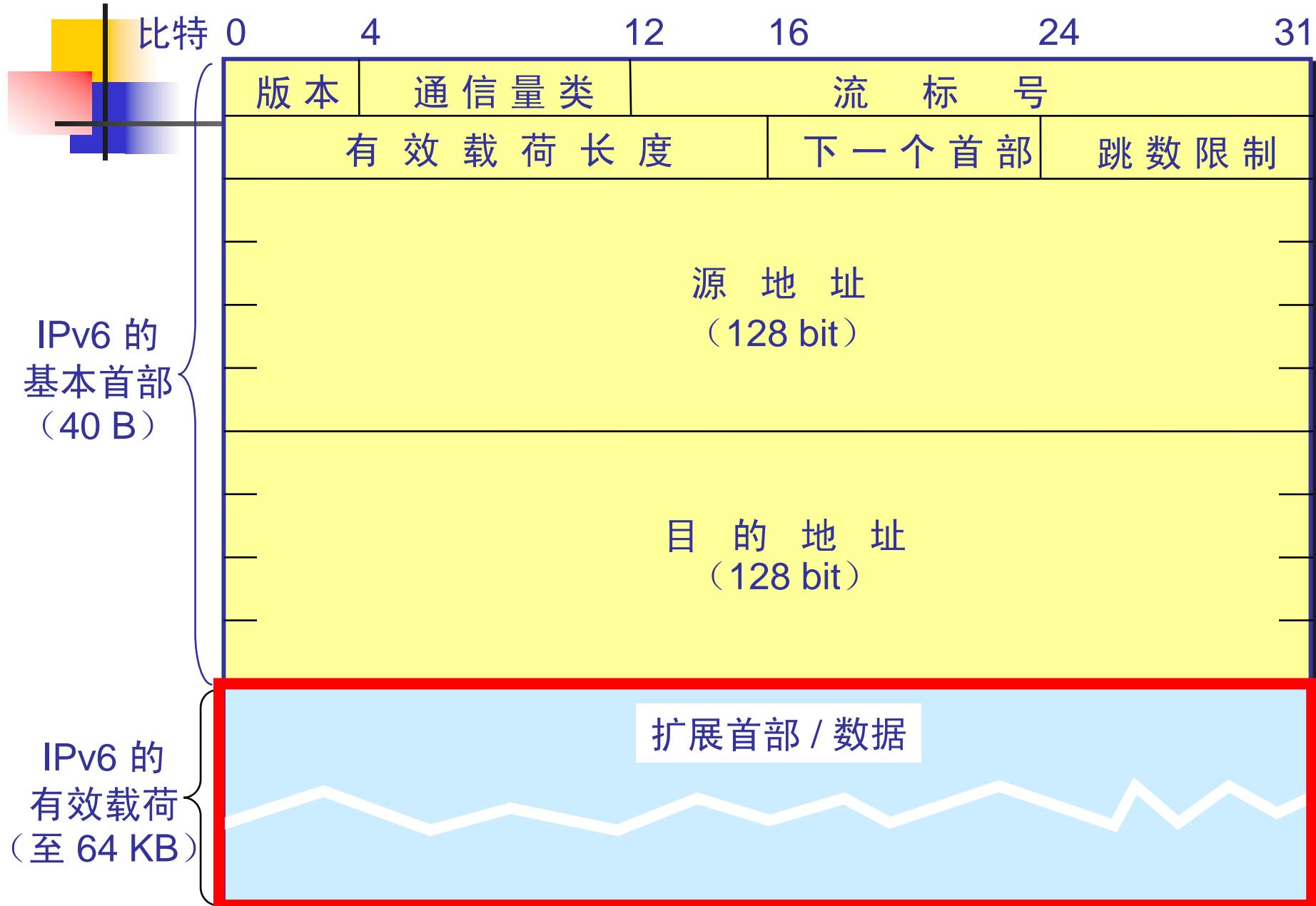
IPv6 数据报的一般形式

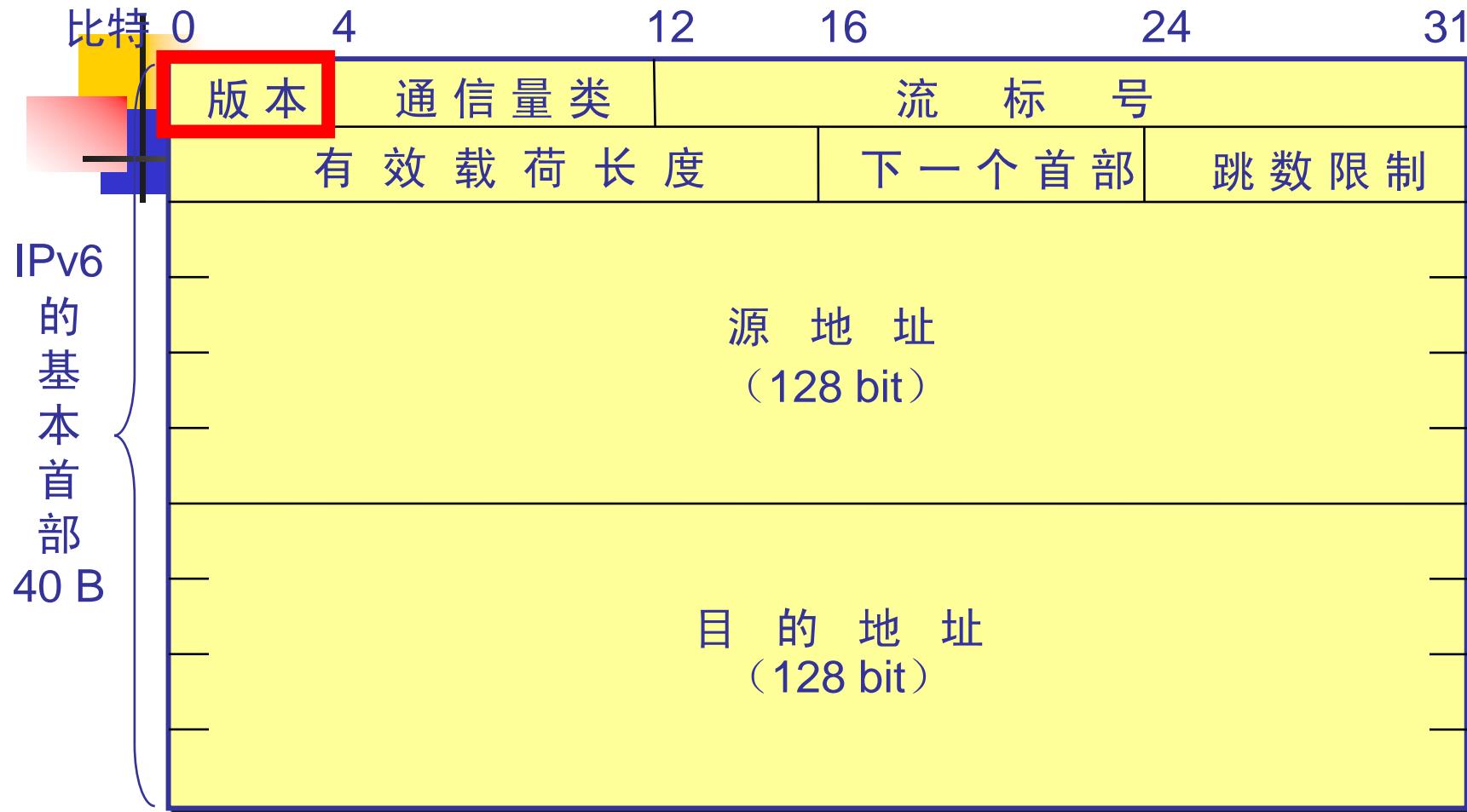


IPv6 数据报头部与 IPv4 数据报头部的对比

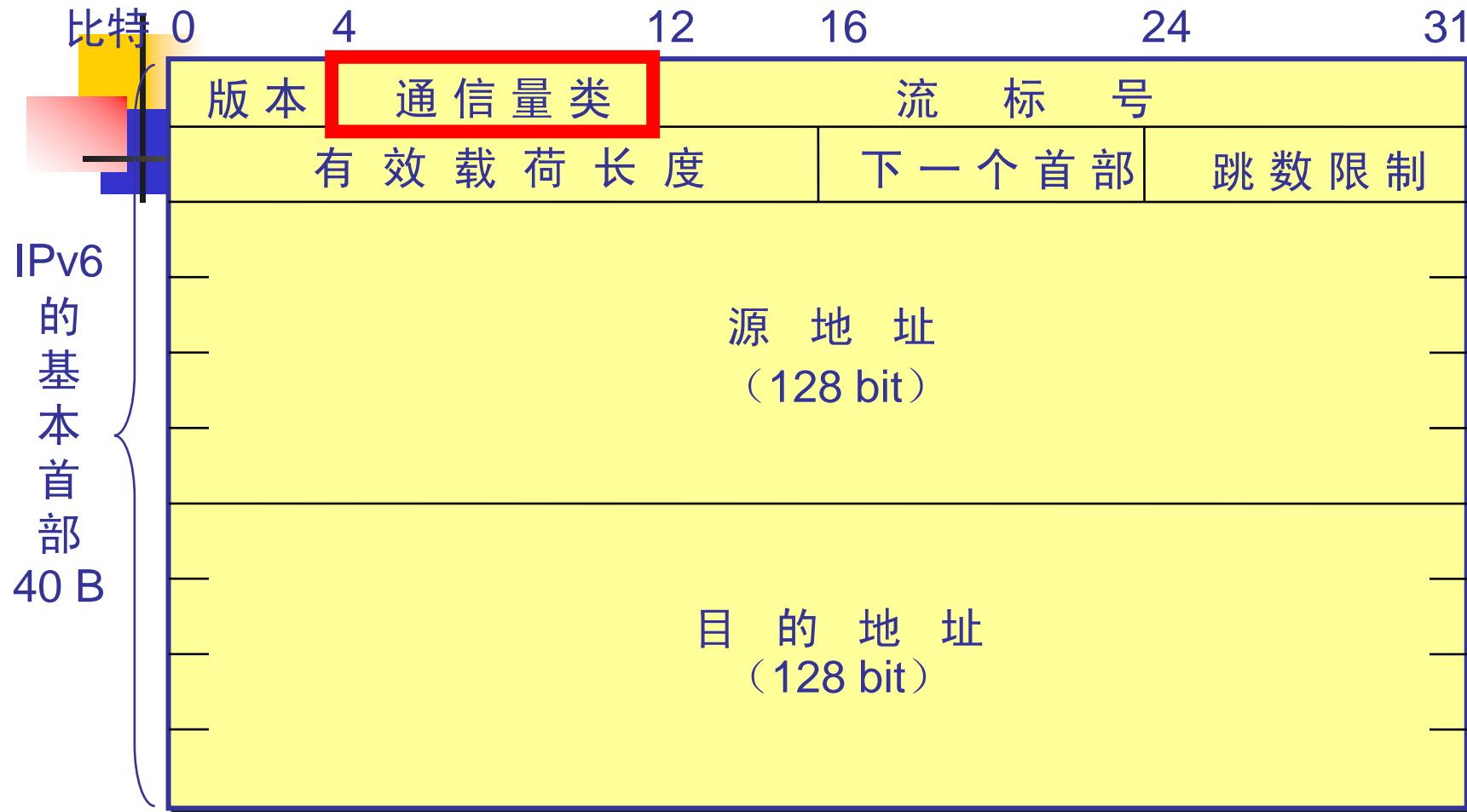




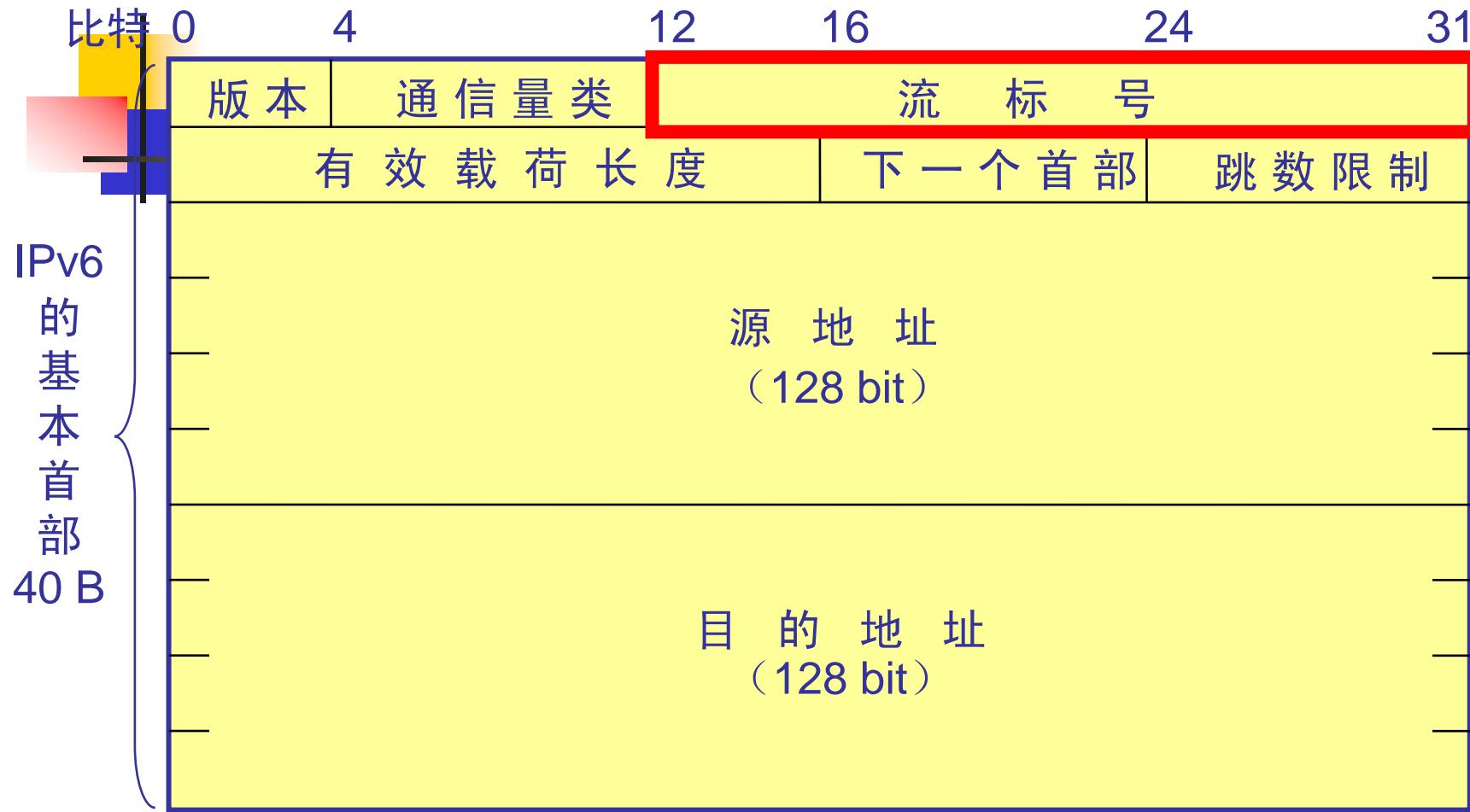




版本(version)—— 4 bit。它指明了协议的版本，对 IPv6 该字段总是 6。

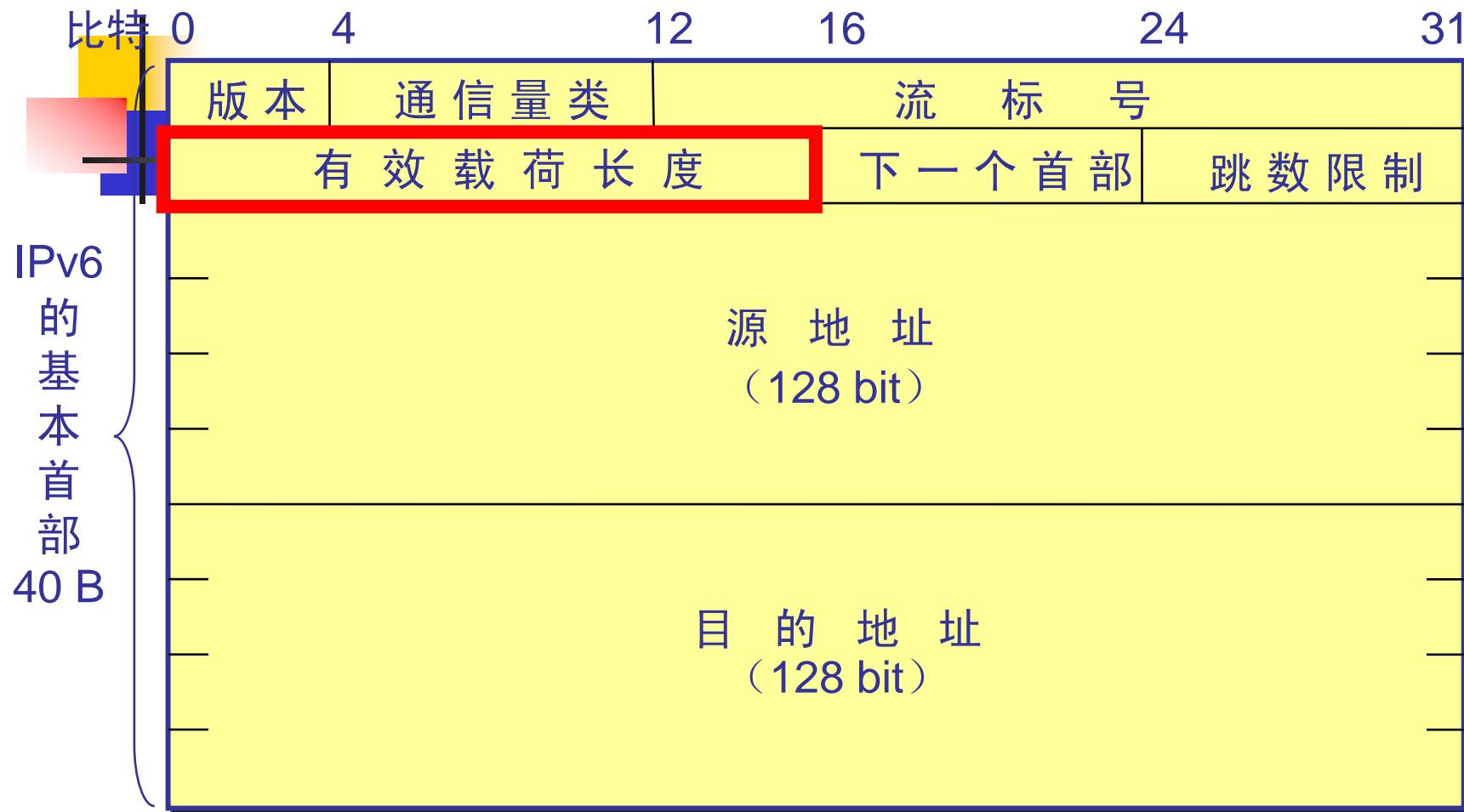


通信量类(traffic class)—— 8 bit。这是为了区分不同的 IPv6 数据报的类别或优先级。目前正在
进行不同的通信量类性能的实验。

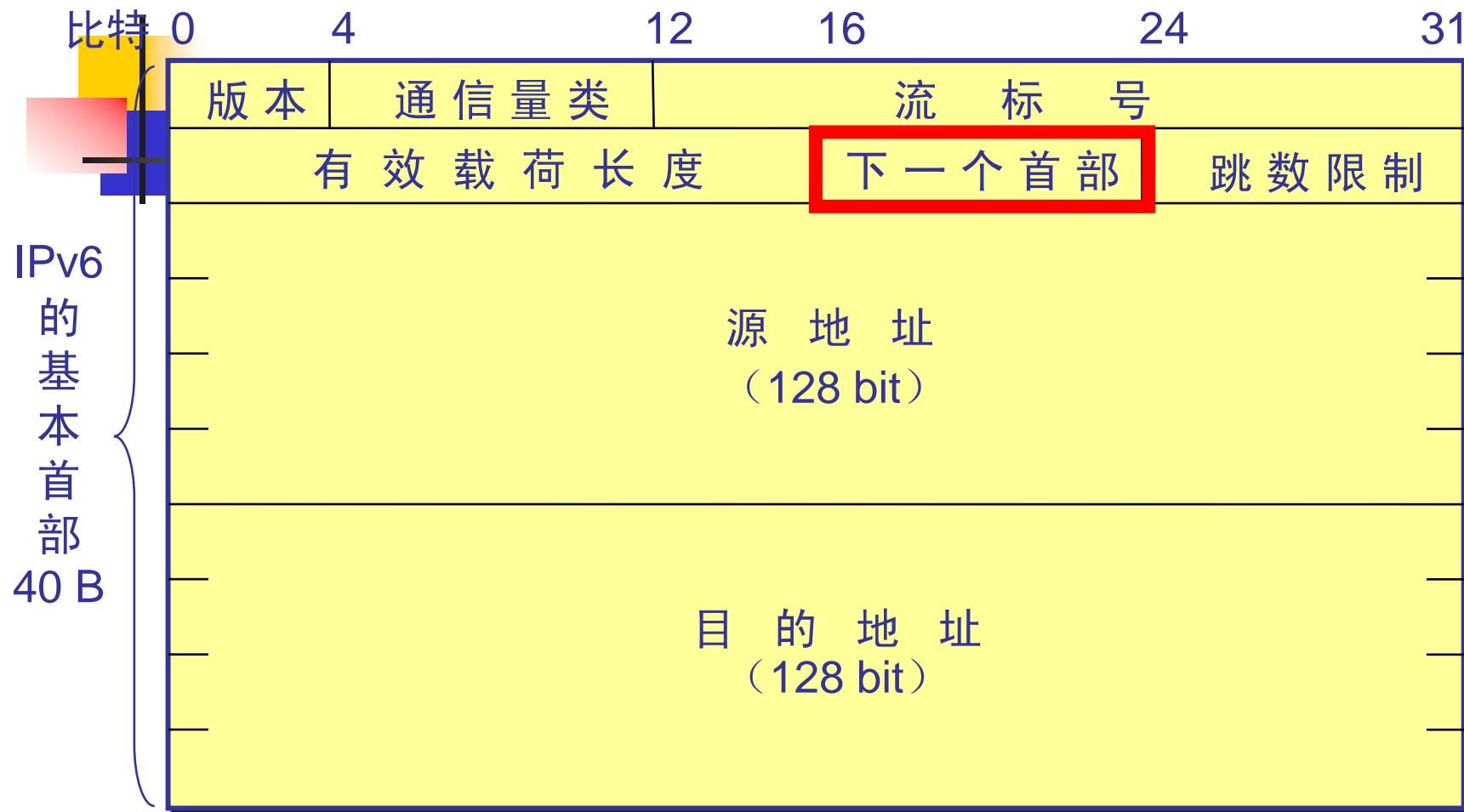


流标号(flow label)—— 20 bit。“流”是互联网络上从特定源点到特定终点的一系列数据报，“流”所经过的路径上的路由器都保证指明的服务质量。

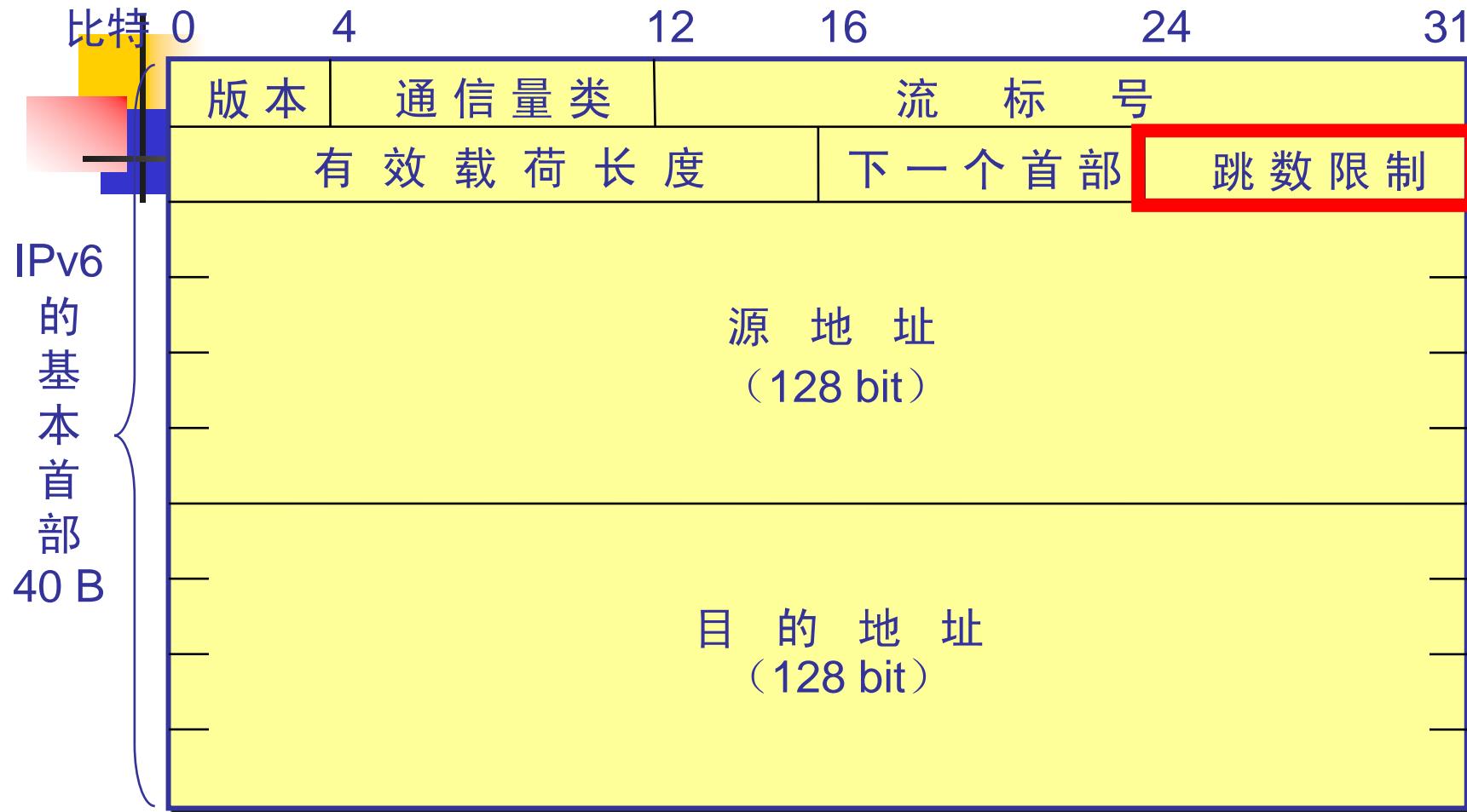
所有属于同一个流的数据报都具有同样的流标号。



有效载荷长度(payload length)—— 16 bit。它指明 IPv6 数据报除基本首部以外的字节数（所有扩展首部都算在有效载荷之内），其最大值是 64 KB。

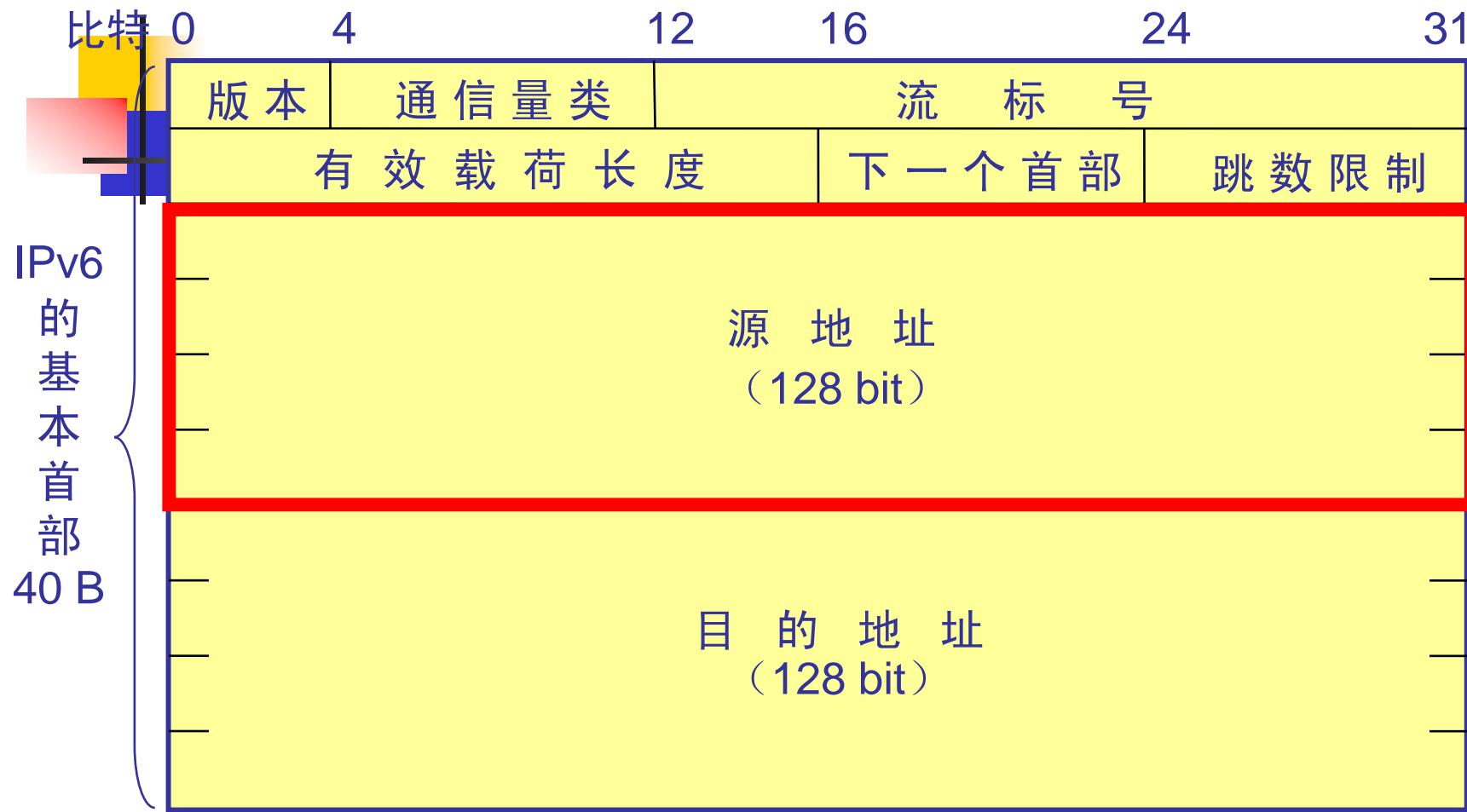


下一个首部(next header)—— 8 bit。它相当于 IPv4 的协议字段或可选字段。

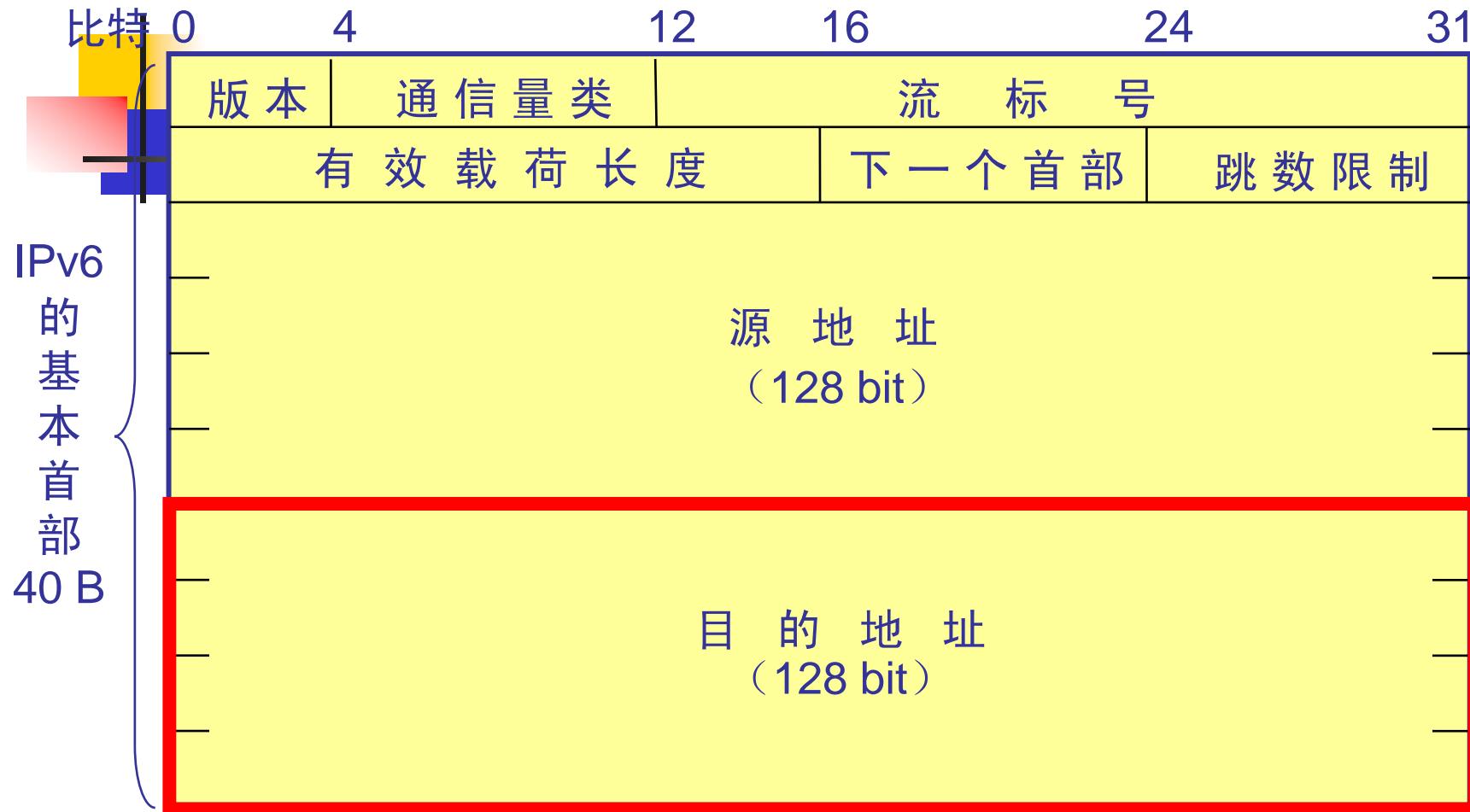


跳数限制(hop limit)——8 bit。源站在数据报发出时即设定跳数限制。路由器在转发数据报时将跳数限制字段中的值减1。

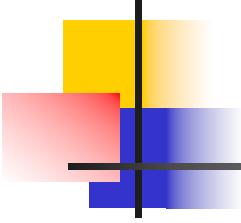
当跳数限制的值为零时，就要将此数据报丢弃。



源地址—— 128 bit。是数据报的发送站的 IP 地址。



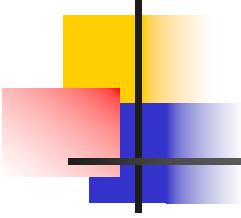
目的地址—— 128 bit。是数据报的接收站的 IP 地址。



6.8.3 IPv6 的扩展首部

1. 扩展首部及下一个首部字段

- IPv6 将原来 IPv4 首部中选项的功能都放在扩展首部中，并将扩展首部留给路径两端的源站和目的站的主机来处理。
- 数据报途中经过的路由器都不处理这些扩展首部（只有一个首部例外，即逐跳选项扩展首部）。
- 这样就大大提高了路由器的处理效率。



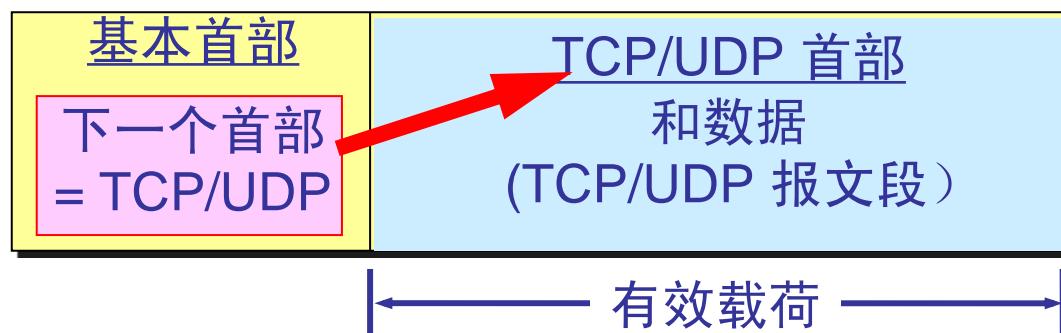
六种扩展首部

在[RFC 2460]中定义了六种扩展首部：

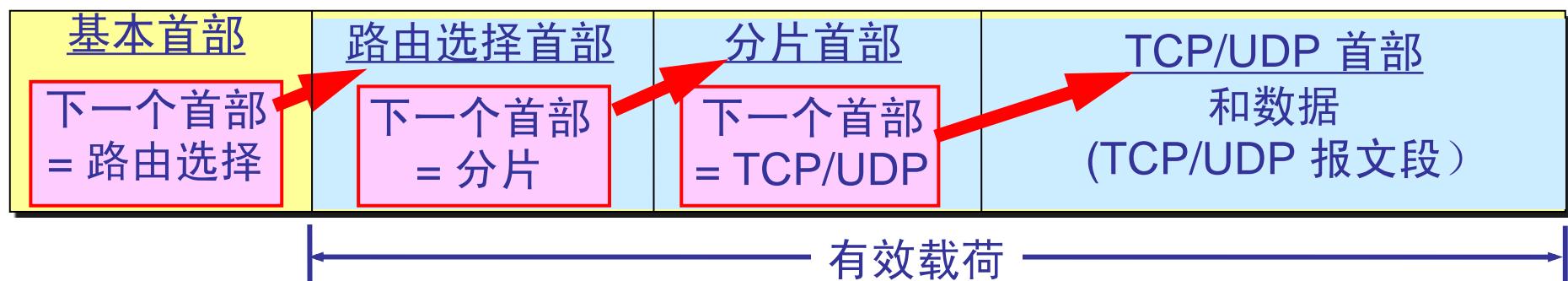
- 逐跳选项
- 路由选择
- 分片
- 鉴别
- 封装安全有效载荷
- 目的站选项

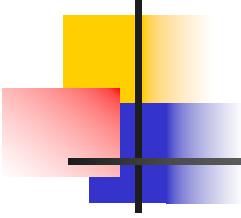
IPv6 的扩展首部

无扩展首部



有扩展首部

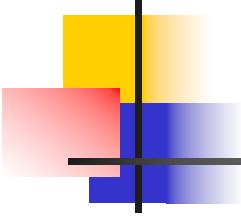




2. 扩展首部举例

- IPv6 将分片限制为由源站来完成。源站可以采用保证的最小 MTU（1280字节），或者在发送数据前完成路径最大传送单元发现(Path MTU Discovery)，以确定沿着该路径到目的站的最小 MTU。
- 分片扩展首部的格式如下：

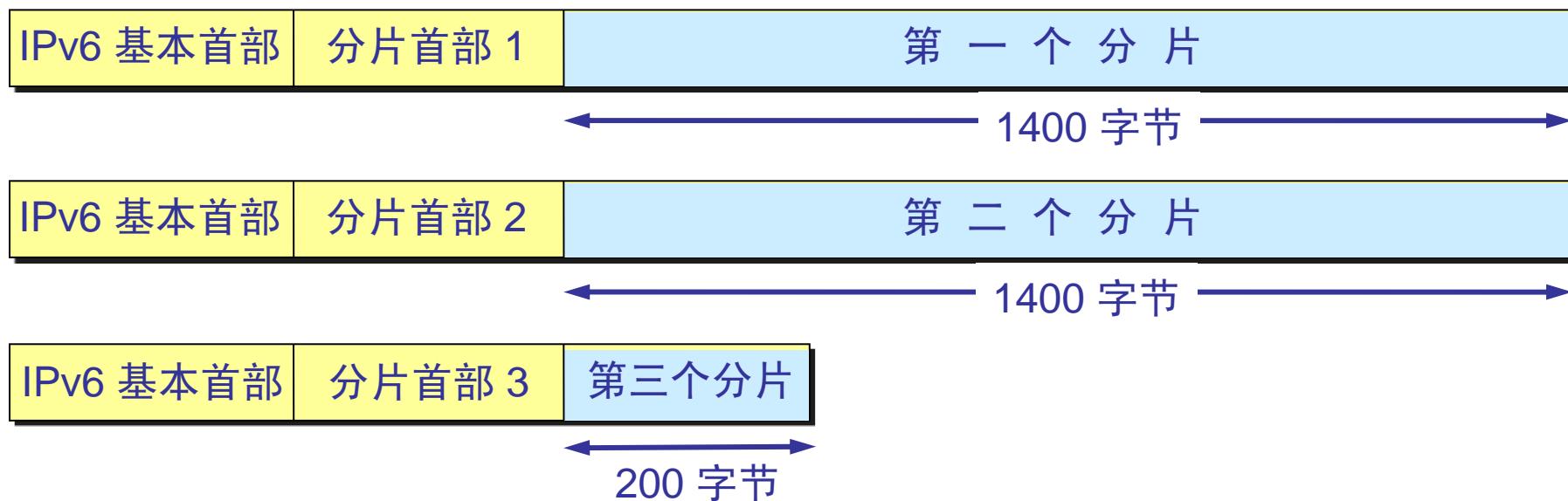
比特 0	8	16	29	31
下一个首部	保 留	片 偏 移	保 留	M
标 识 符				

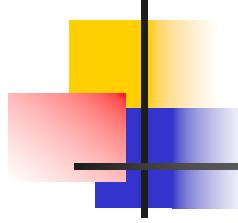


扩展首部举例

- IPv6 数据报的有效载荷长度为 3000 字节。下层的以太网的最大传送单元 MTU 是 1500 字节。
- 分成三个数据报片，两个 1400 字节长，最后一个 200 字节长。

扩展首部





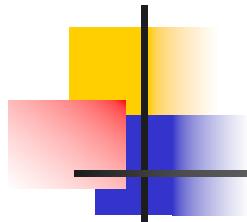
6.8.4 IPv6 的地址空间

1. 128 bit 的地址空间

IPv6 数据报的目的地址可以是以下三种基本类型地址之一：

- (1) **单播(unicast)** 单播就是传统的点对点通信。
- (2) **多播(multicast)** 多播是一点对多点的通信。
- (3) **任播(anycast)** 这是 IPv6 增加的一种类型。

任播的目的站是一组计算机，但数据报在交付时只交付给其中的一个，通常是距离最近的一个。

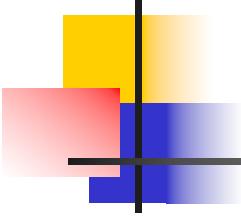


冒号十六进制记法 (colon hexadecimal notation)

- 每个 16 bit 的值用十六进制值表示，各值之间用冒号分隔。

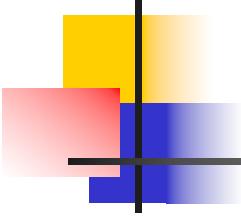
68E6:8C64:FFFF:FFFF:0:1180:960A:FFFF

- 零压缩(zero compression)，即一连串连续的零可以为一对冒号所取代。
- FF05:0:0:0:0:0:B3 可以写成：
- FF05::B3



点分十进制记法的后缀

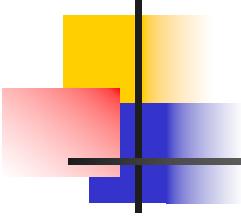
- 0:0:0:0:0:128.10.2.1
再使用零压缩即可得出： ::128.10.2.1
- CIDR 的斜线表示法仍然可用。
- 60 bit的前缀 12AB00000000CD3 可记为：
12AB:0000:0000:CD30:0000:0000:0000:0000/60
或12AB::CD30:0:0:0:0/60
或12AB:0:0:CD30::/60



2. 地址空间的分配

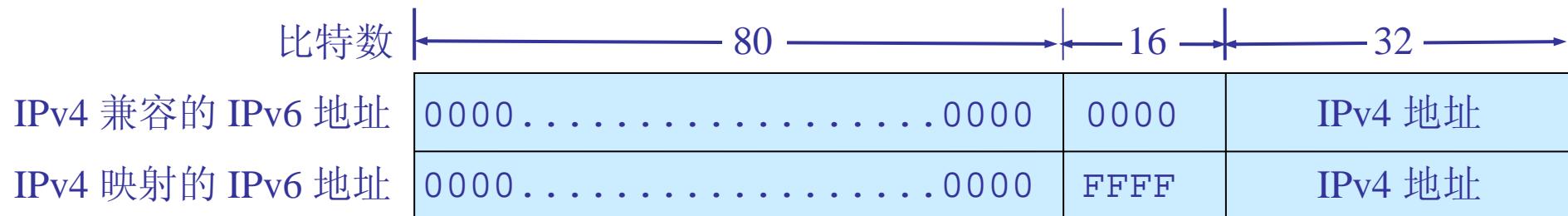
- IPv6 将 128 bit 地址空间分为两大部分。
 - 第一部分是可变长度的类型前缀，它定义了地址的目的。
 - 第二部分是地址的其余部分，其长度也是可变的。





前缀为 0000 0000 的地址

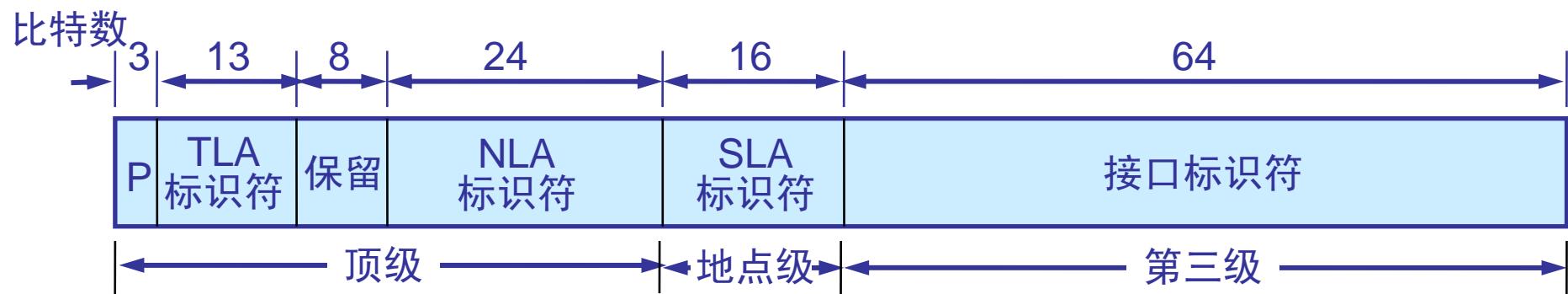
- 前缀为 0000 0000 是保留一小部分地址与 IPv4 兼容的，这是因为必须要考虑到在比较长的时期 IPv4 和 IPv6 将会同时存在，而有的结点不支持 IPv6。
- 因此数据报在这两类结点之间转发时，就必须进行地址的转换。

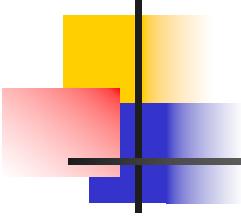


IPv6 单播地址的等级结构

IPv6 扩展了地址的分级概念，使用以下三个等级：

- (1) 第一级（顶级），指明全球都知道的公共拓扑。
- (2) 第二级（地点级），指明单个的地点。
- (3) 第三级，指明单个的网络接口。

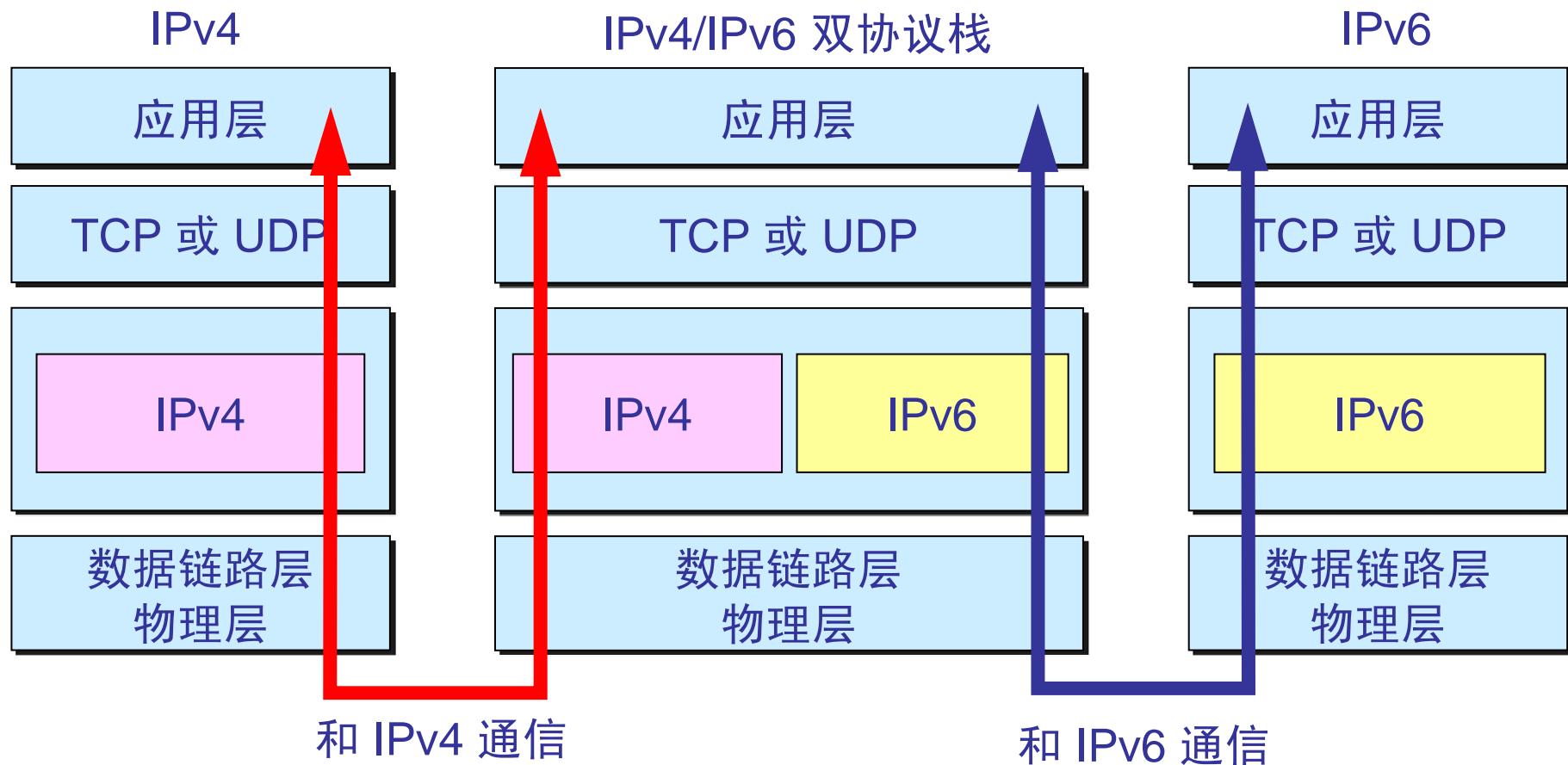




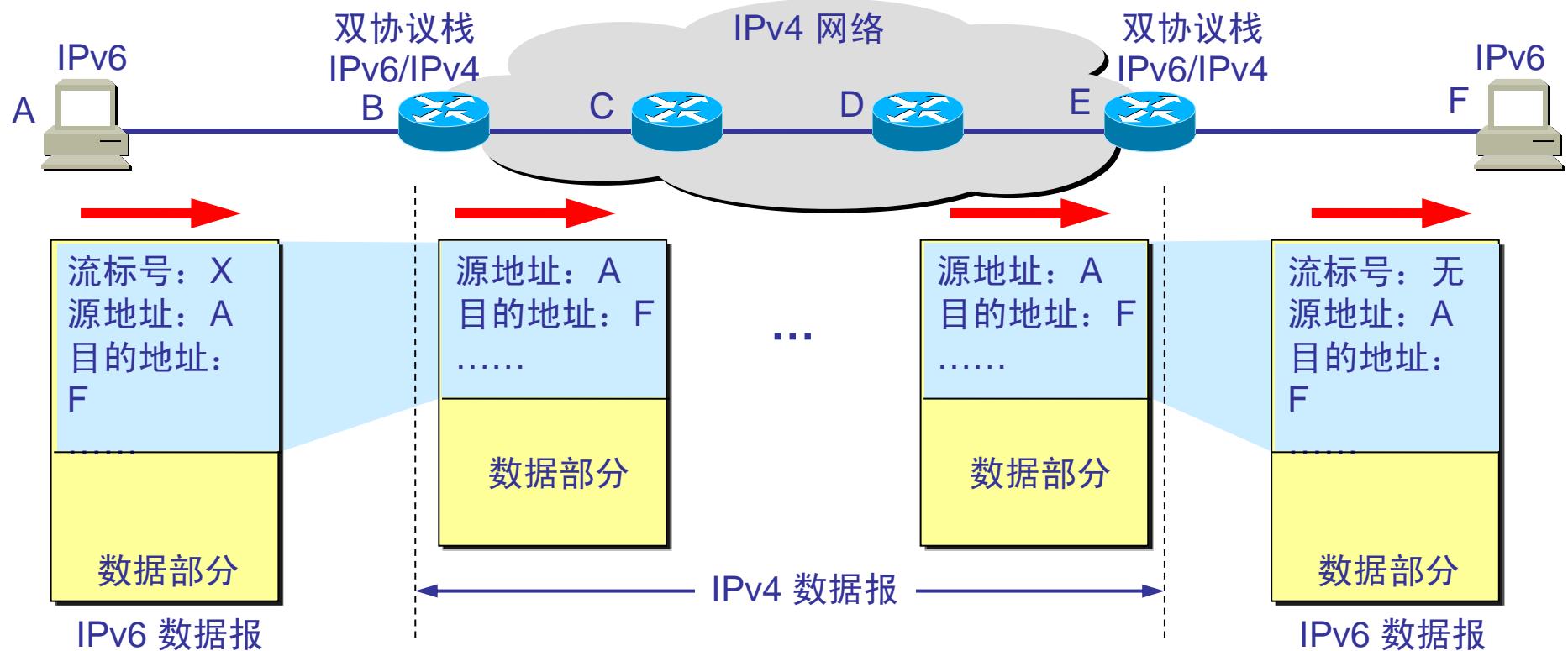
6.8.5 从 IPv4 向 IPv6 过渡

- 向 IPv6 过渡只能采用逐步演进的办法，同时，还必须使新安装的 IPv6 系统能够向后兼容。
- IPv6 系统必须能够接收和转发 IPv4 分组，并且能够为 IPv4 分组选择路由。
- 双协议栈(dual stack)是指在完全过渡到 IPv6 之前，使一部分主机（或路由器）装有两个协议栈，一个 IPv4 和一个 IPv6。

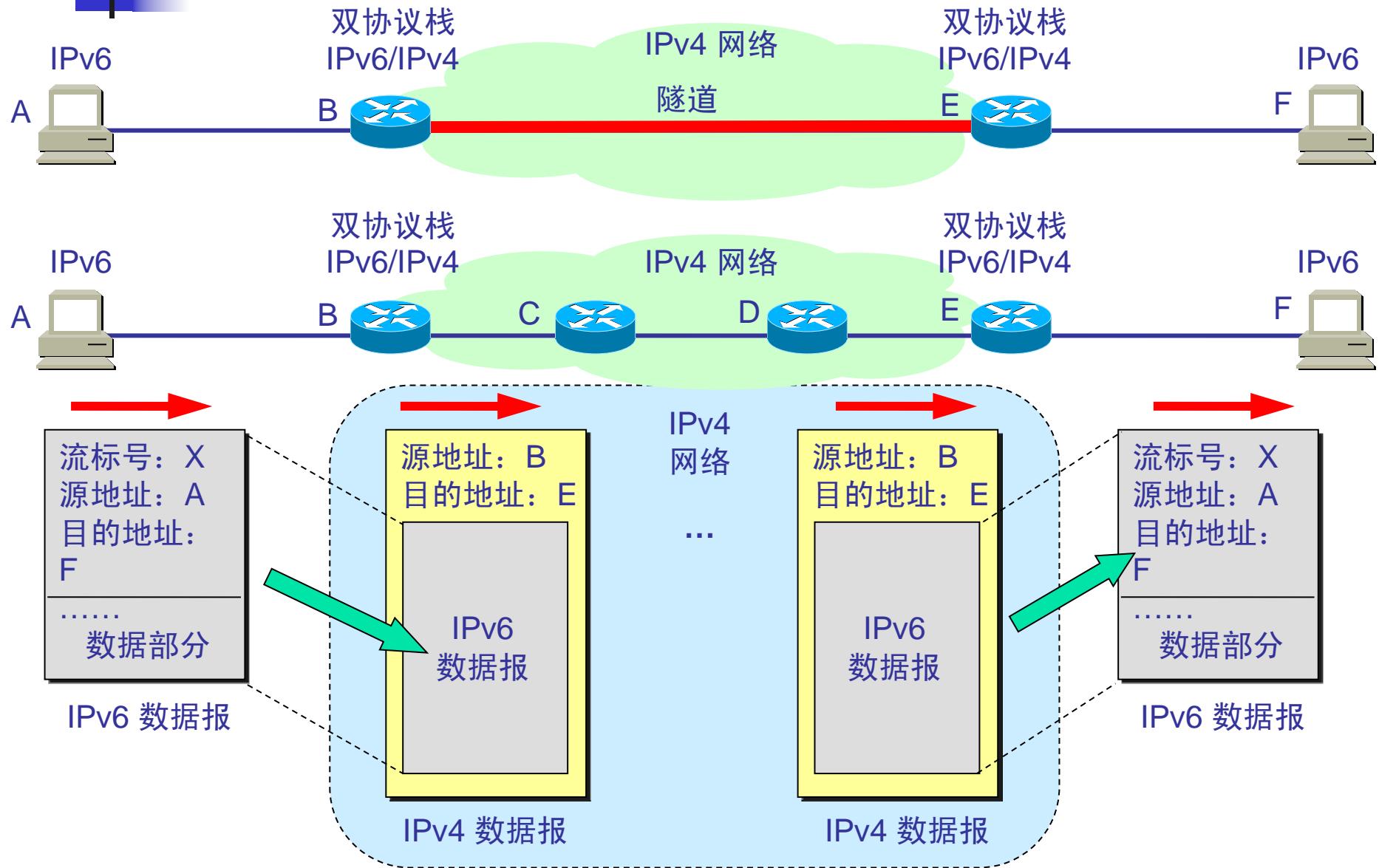
双协议栈

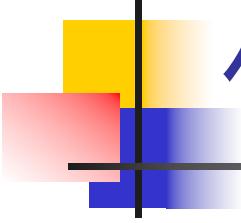


用双协议栈进行 从 IPv4 到 IPv6 的过渡



使用隧道技术从 IPv4 到 IPv6 过渡





作业

4-4, 5, 7, 9, 10, 13, 17, 20, 21, 22,
26, 28, 29, 30, 41