计 算 机 网 络

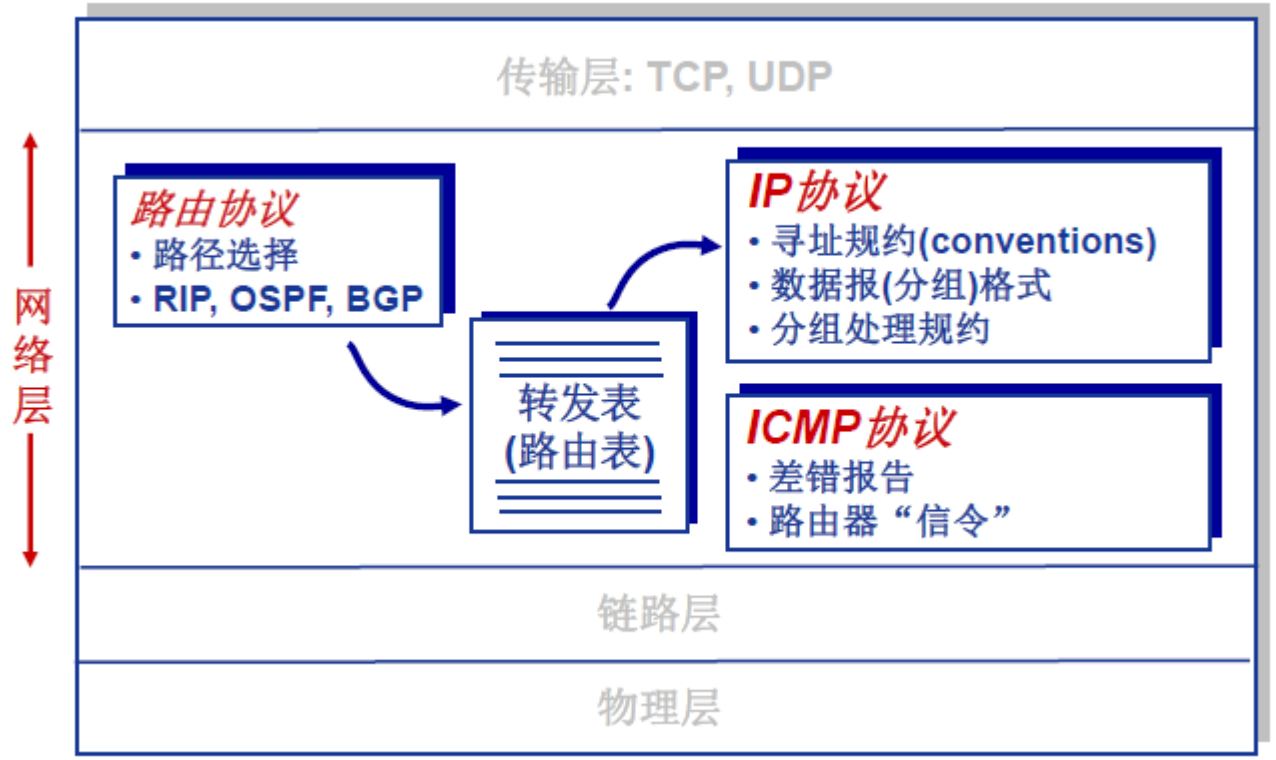
西北工业大学 软件与微电子学院

计算机网络

第5章 网络层-控制平面

网络层

主机、路由器网络层主要功能：



因特网的路由选择协议

1. 理想的路由算法

- 算法必须是正确的和完整的。
- 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，这就是说，要有自适应性。
- 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
  - 它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。

路由策略

- 静态路由选择策略——即非自适应路由选择，其特点是简单和开销较小，但不能及时适应网络状态的变化。
- 动态路由选择策略——即自适应路由选择，其特点是能较好地适应网络状态的变化，但实现起来较为复杂，开销也比较大。

路由策略

- 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。

路由策略

- 自治系统 AS 的定义：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，同时还使用一种 AS 之间的路由选择协议用以确定分组在 AS之间的路由。
- 现在对自治系统 AS 的定义是强调下面的事实：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 表现出的是一个单一的和一致的路由选择策略。

两大类路由协议

- 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。

两大类路由协议

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)

两大类路由协议

- 内部网关协议 IGP：具体的协议有多种，如 RIP 和 OSPF 等。
- 外部网关协议 EGP：目前使用的协议就是 BGP。

路由算法

网络的数学模型：图

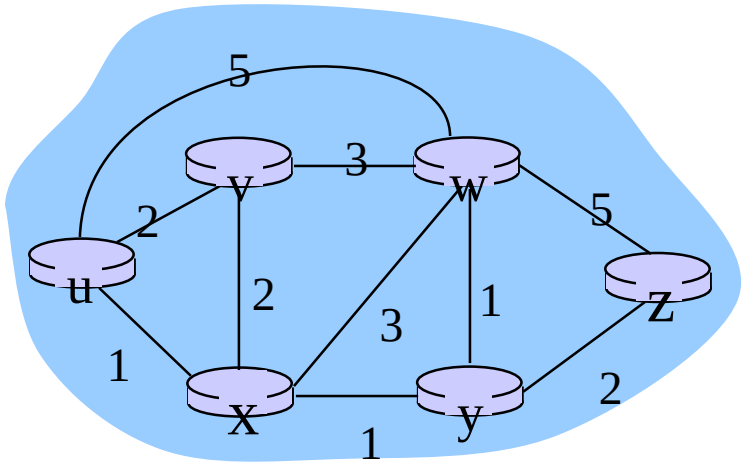


图: G = (N, E)

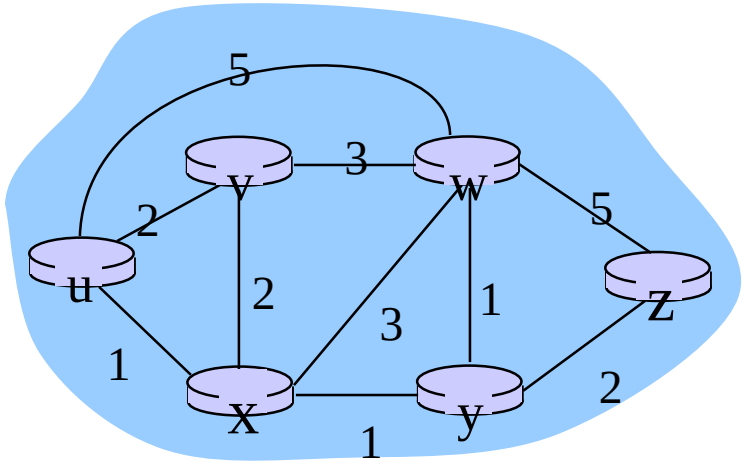
N = 路由器集合= { u, v, w, x, y, z }

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

附注: 图的抽象在网络领域应用很广泛

E.g.:P2P，其中，N是 peers集合，而E是TCP连接集合

图抽象：费用（Cost）



c(x, x’) = 链路**(x, x’)**的费用

e.g., c(w, z) = 5

每段链路的费用可以总是**1**， 或者是带宽的倒数、拥塞程度等

路径费用：**(x1, x2, x3,…, xp) = c(x1,x2) + c(x2,x3) + … + c(xp-1,xp)**

关键问题**:** 源到目的（如**u**到**z**）的最小费用路径是什么?

路由算法**:** 寻找最小费用路径的算法

路由算法分类

静态路由 **vs** 动态路由?

静态路由:

- 手工配置
- 路由更新慢
- 优先级高

动态路由:

- 路由更新快
- 定期更新
- 及时响应链路费用或网络拓扑变化

全局信息 **vs** 分散信息?

全局信息:

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

分散(decentralized)信息:

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

路由算法分类

Dijkstra 算法

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

符号:

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

典型的链路状态路由算法：Dijkstra 算法

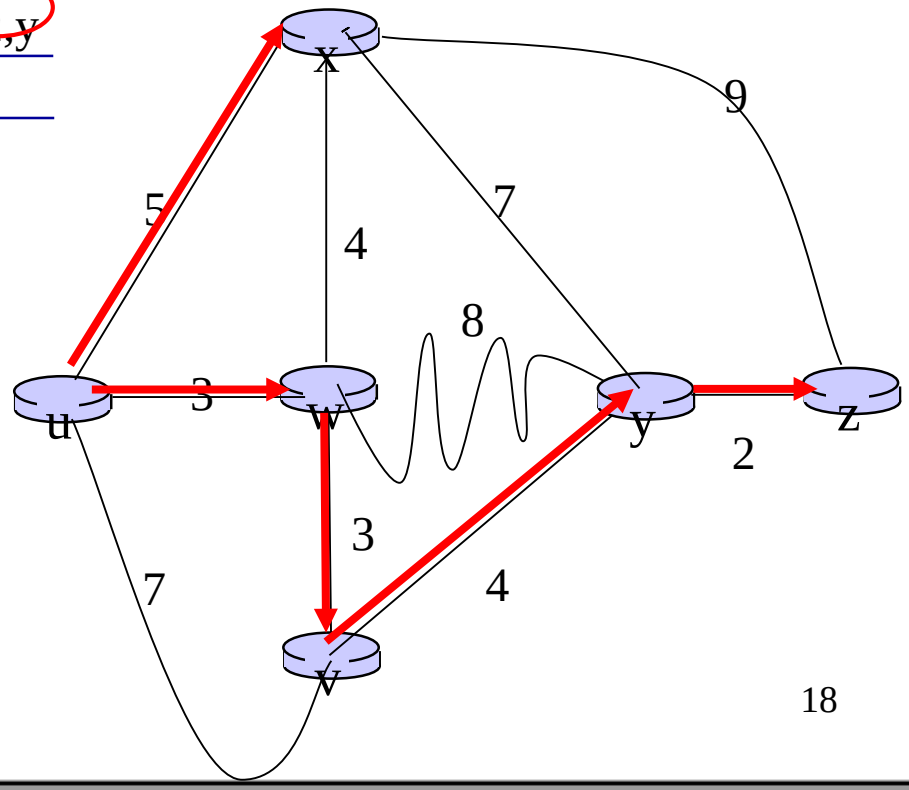
```
1  初始化:
2   N' = {u}
3   for 所有结点v
4     if v毗邻u
5        then D(v) = c(u,v)
6     else D(v) = ∞
7
8  Loop
9    找出不在 N'中的w ， 满足D(w)最小
10   将w加入N'
11   更新w的所有不在N'中的邻居v的D(v) :
12     D(v) = min( D(v), D(w) + c(w,v) )
13   /*到达v的新费用或者是原先到达v的费用，或者是
14     已知的到达w的最短路径费用加上w到v的费用 */
15 until 所有结点在N'中
```

Dijkstra 算法

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

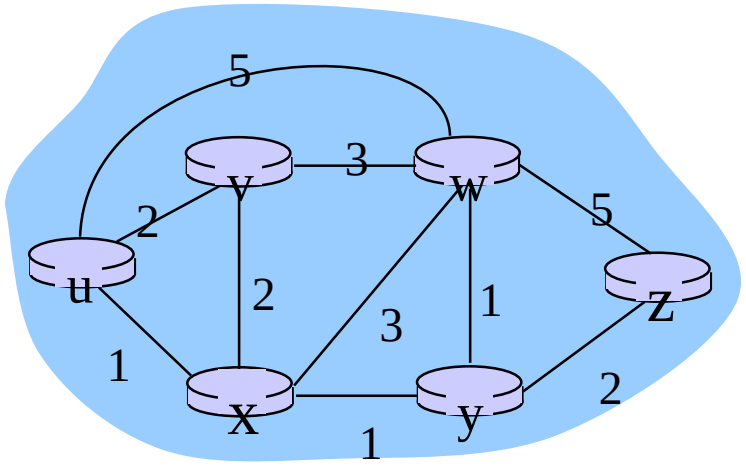
注意:

新的最短路径一定是从某条已知的最短路径衍生出来



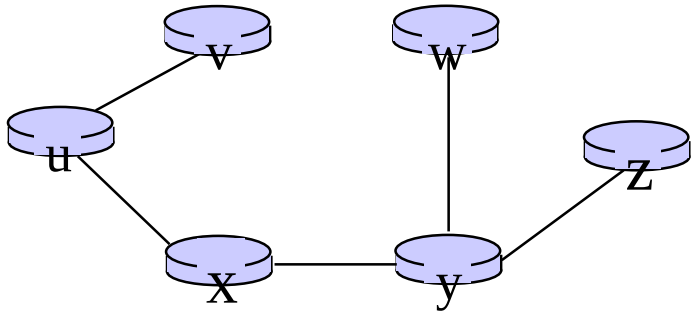
Dijkstra 算法

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					



Dijkstra 算法

u的最终最短路径树:



u的最终转发表

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

链路状态路由算法

- 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。

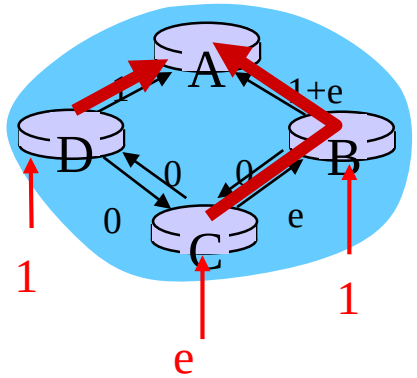
链路状态算法

算法复杂性: n个结点

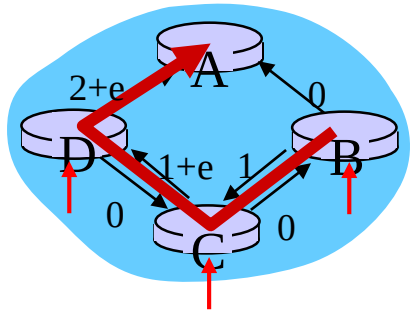
- 每次迭代: 需要检测所有不在集合N'中的结点w
- n(n+1)/2次比较: O(n^2)
- 更高效的实现: O(nlogn)

存在震荡(oscillations)可能:

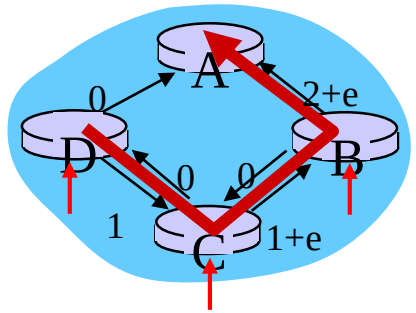
- e.g., 假设链路费用是该链路承载的通信量:



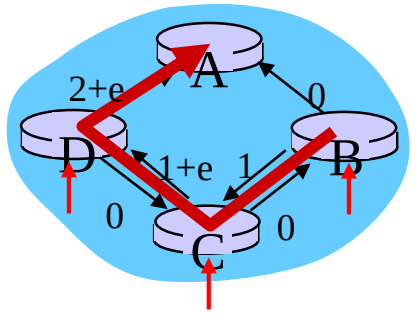
初始



基于这些费用
计算新的路由



基于这些费用
计算新的路由



基于这些费用
计算新的路由

距离向量(Distance Vector)路由算法

Bellman-Ford方程(动态规划)

令:

dx(y):=从x到y最短路径的费用（距离）

则:

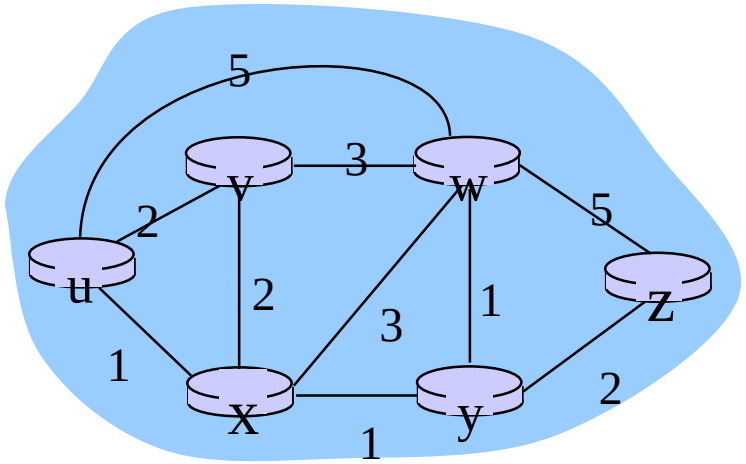
$$d_x(y) = \min_v \{c(x,v) + d_v(y)\}$$

v：遍历所有的邻居v

c(x,v)：x到邻居v的费用

$d_v(y)$：从邻居v到达目的y的费用

距离向量(Distance Vector)路由算法



显然: **dv(z) = 5, dx(z) = 3, dw(z) = 3**

du(z) = min { c(u,v) + dv(z),

c(u,x) + dx(z),

c(u,w) + dw(z) }

= min {2 + 5,

1 + 3,

5 + 3} = 4

重点：结点获得最短路径的下一跳, 该信息用于转发表中!

距离向量(Distance Vector)路由算法

Dx(y) = 从结点x到结点y的最小费用估计

- x维护距离向量(DV): **D**x = [Dx(y): y ϵ N]
- 已知到达每个邻居的费用: c(x,v)
- 维护其所有邻居的距离向量: **D**v = [Dv(y): y ϵ N]

核心思想:

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

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

- Dx(y)将最终收敛于实际的最小费用 dx(y)

距离向量(Distance Vector)路由算法

异步迭代:

- 引发每次局部迭代的因素

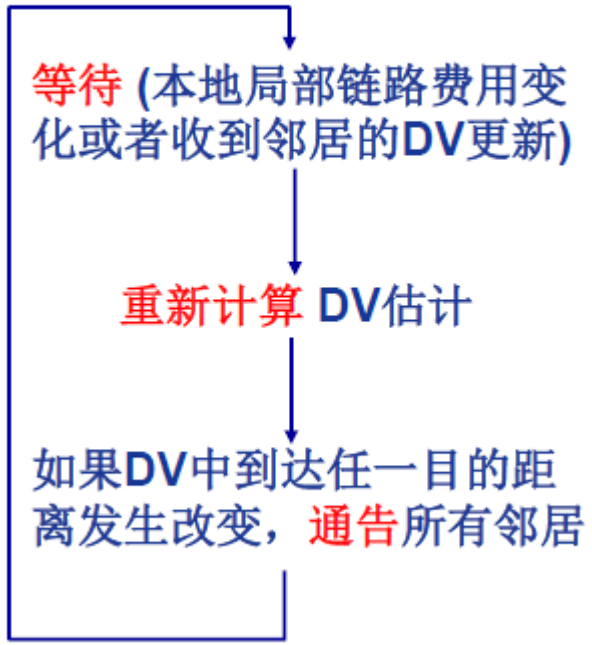
  局部链路费用改变

  来自邻居的DV更新

分布式:

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

每个结点：



距离向量(Distance Vector)路由算法

node x table

from \ cost to	x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

node y table

from \ cost to	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

node z table

from \ cost to	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0

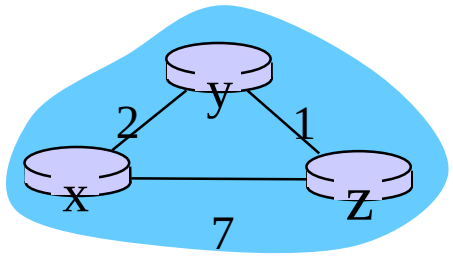
from \ cost to	x	y	z
x	0	2	3
y	2	0	1
z	7	1	0

$D_x(y) = \min\{2, c(x,y) + D_y(y), c(x,z) + D_z(y)\}$

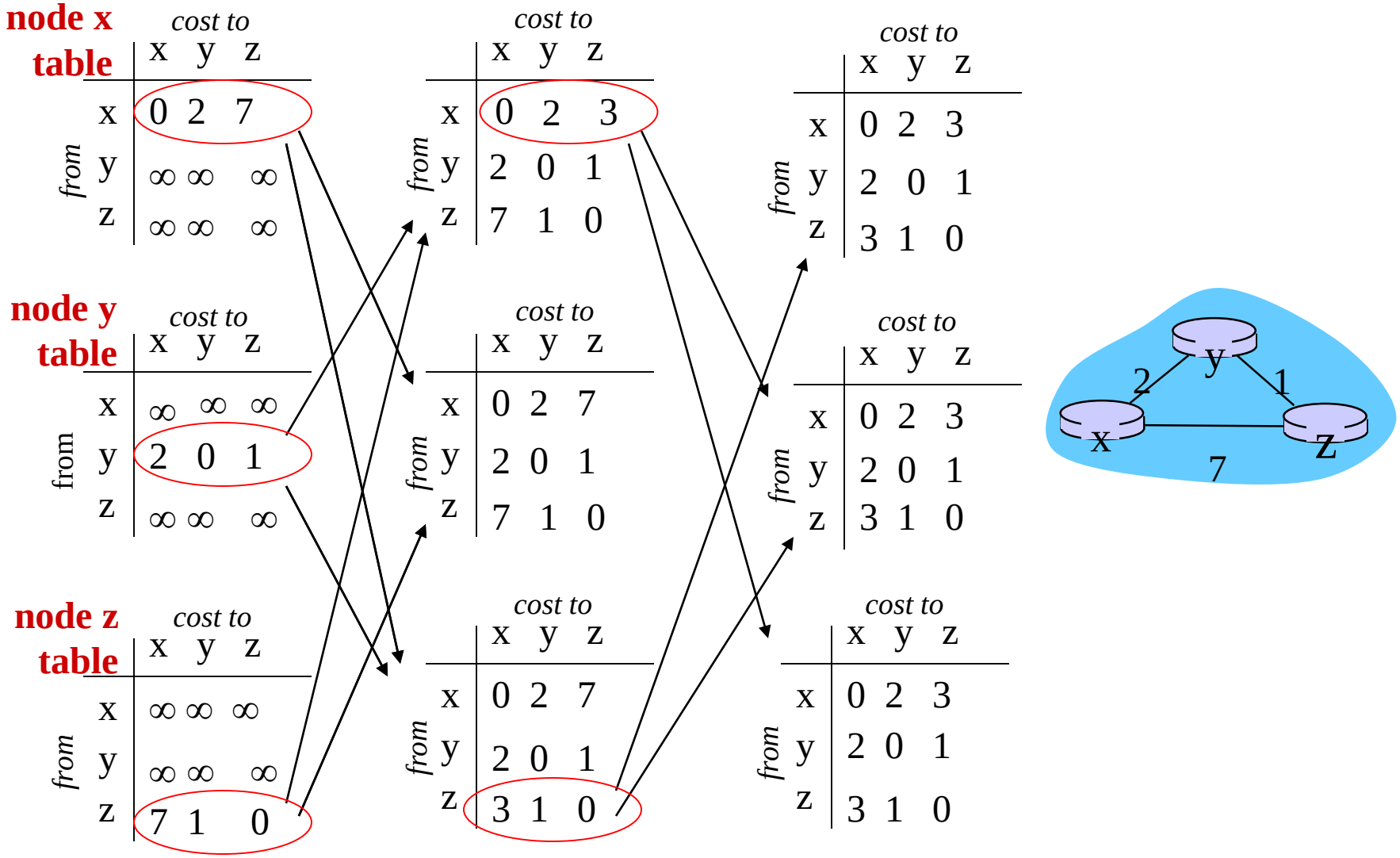
$= \min\{2, \ 2+0 \ , 7+1\} = 2$

$D_x(z) = \min\{7, \ c(x,y) + D_y(z), c(x,z) + D_z(z)\}$

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



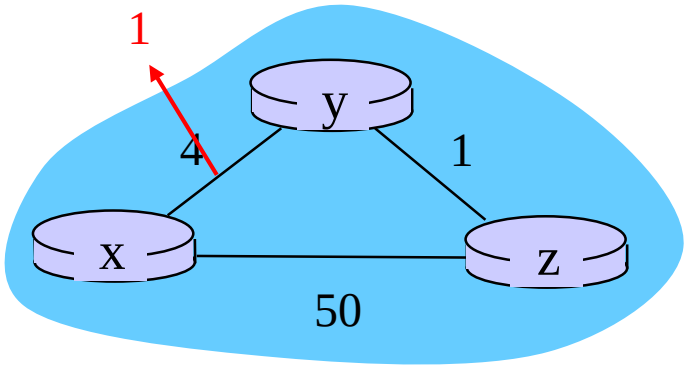
距离向量(Distance Vector)路由算法



距离向量(Distance Vector)路由算法

链路费用变化：

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



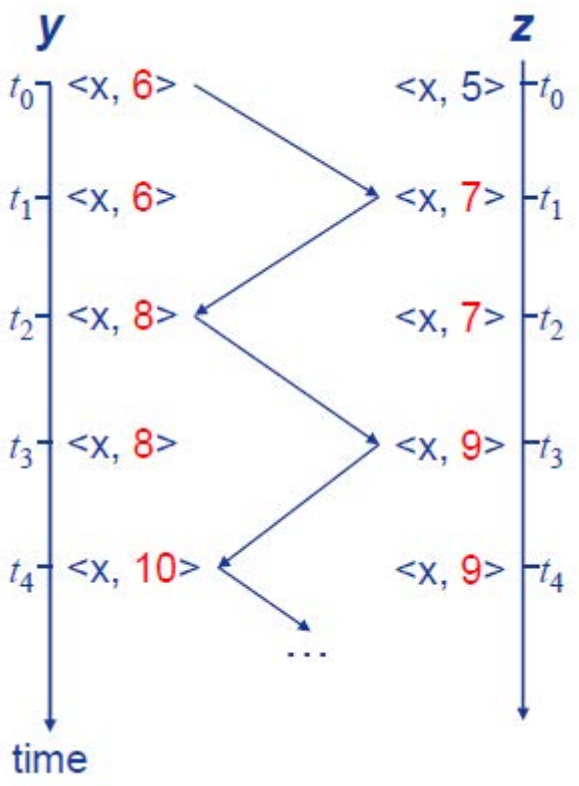
t0 **：*y*检测到链路费用改变，更新DV，通告其邻居.**

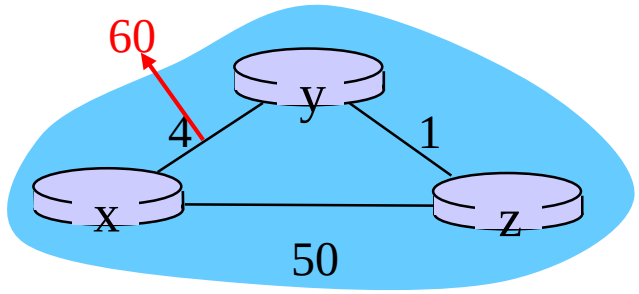
t1 **：*z*收到*y*的DV更新，更新其距离向量表，计算到达*x*的最新最小费用，更新其DV，并发送给其所有邻居.**

t2 **：*y*收到*z*的DV更新，更新其距离向量表，重新计算*y*的DV，未发生改变，不再向*z*发送*DV*.**

“☺好消息传播快！”　　“坏消息会怎么样呢？”

距离向量(Distance Vector)路由算法





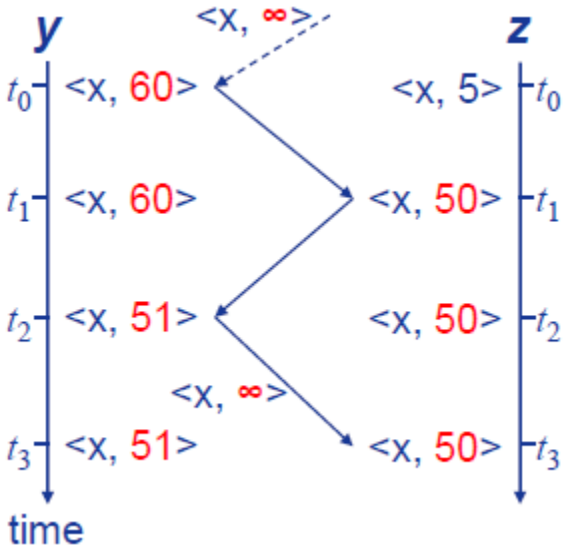
坏消息传播慢！

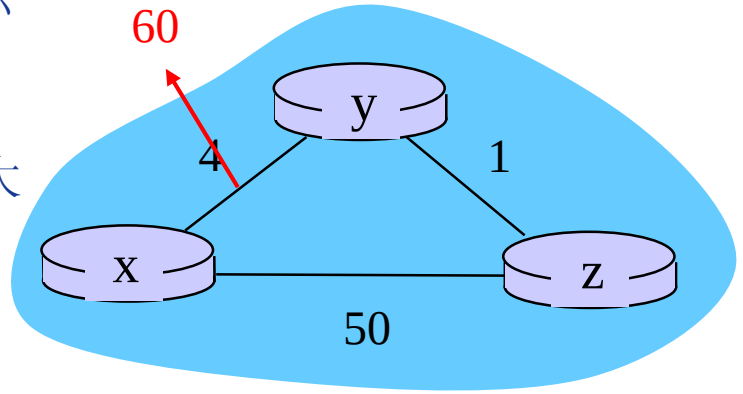
— “无穷计数 (count to infinity)” 问题！

距离向量(Distance Vector)路由算法

毒性逆转**(poisoned reverse):**

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



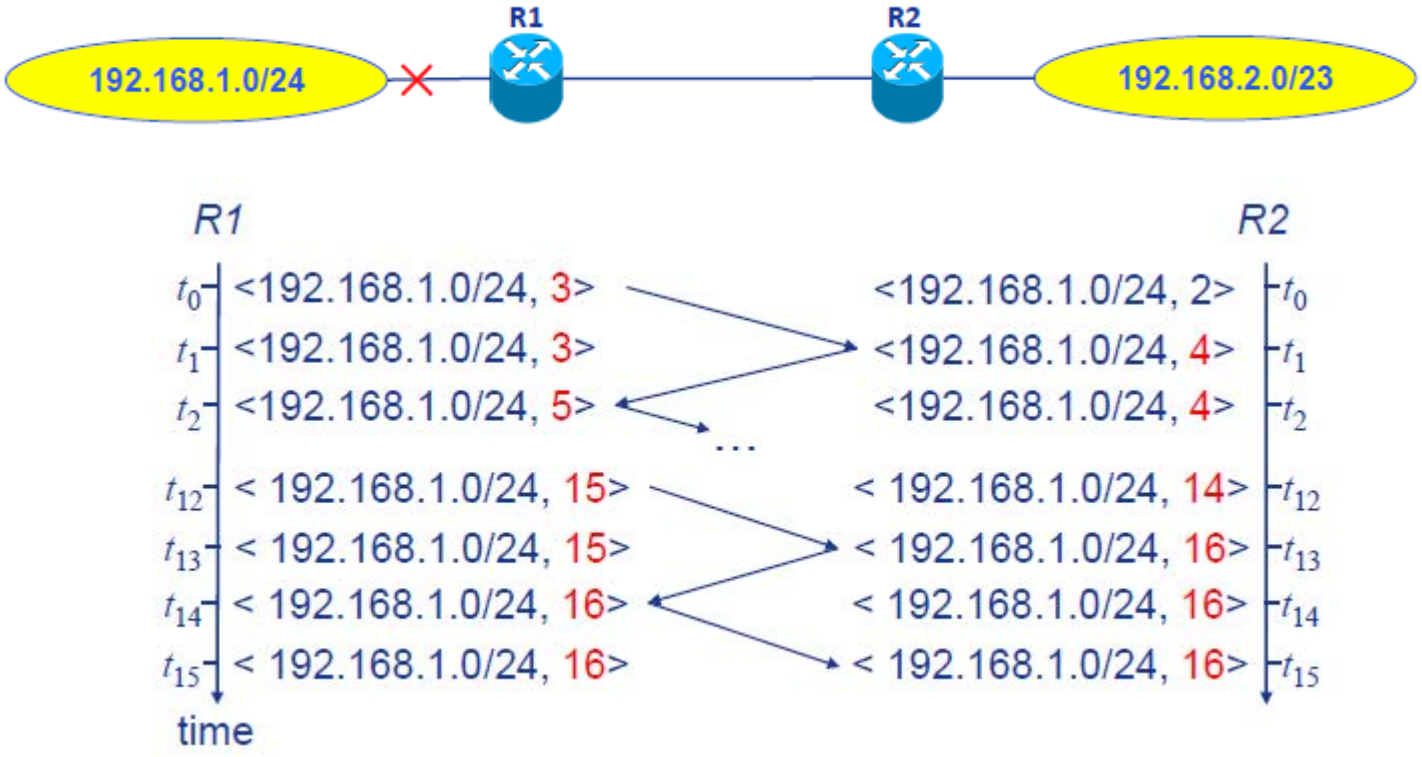


毒性逆转能否彻底解决无穷计数问题?

距离向量(Distance Vector)路由算法

定义最大度量**(maximum metric):**

- 定义一个最大的有效费用值，如15跳步，16跳步表示∞



内部网关协议 RIP路由协议

1. 工作原理

- 路由信息协议 RIP 是内部网关协议 IGP中最先得到广泛使用的协议。
- 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”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”，

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。

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。

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。

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。

内部网关协议 OSPF

1. 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”。

链路状态数据库

- 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。

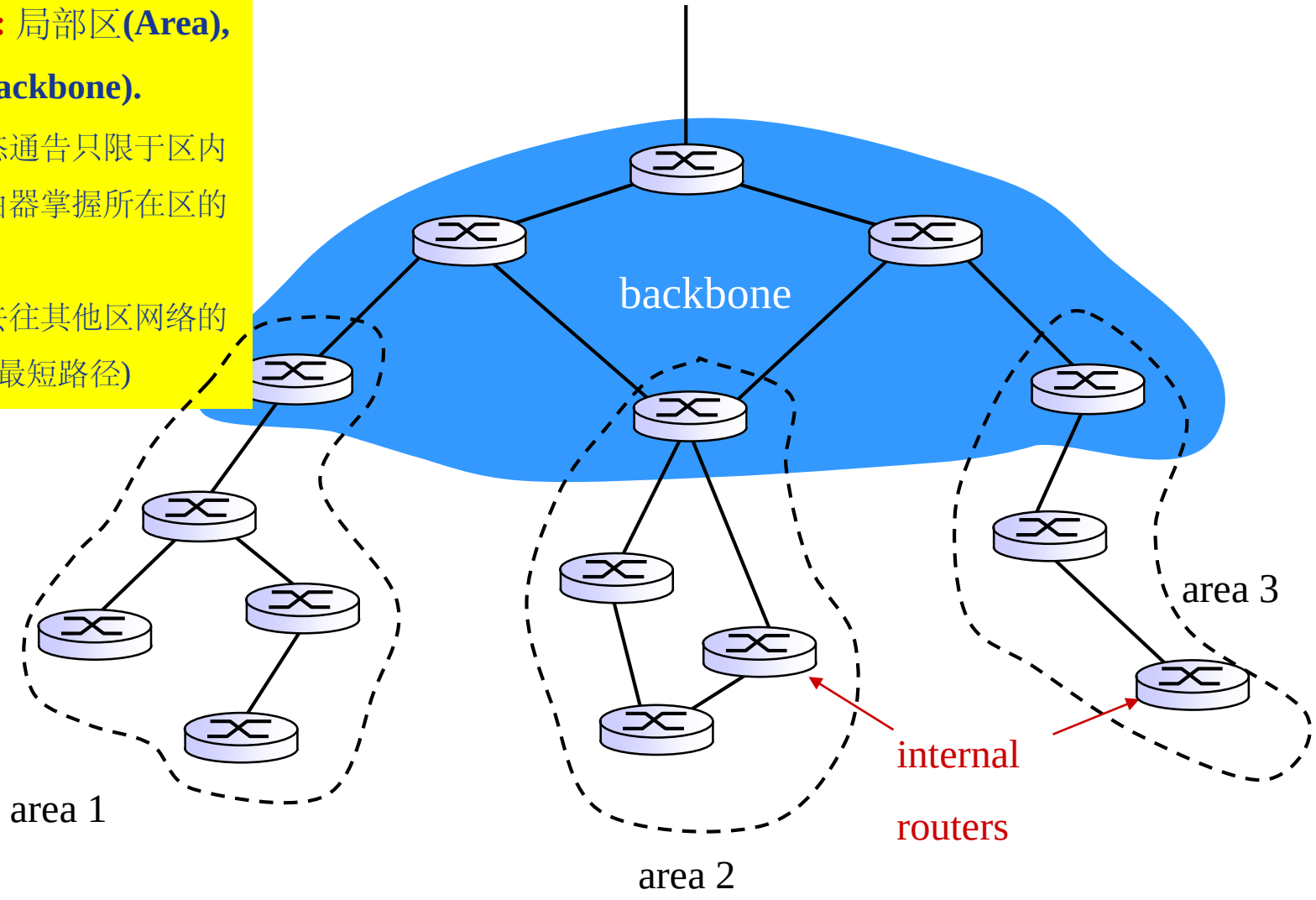
OSPF 的区域划分

- 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。

分层的OSPF

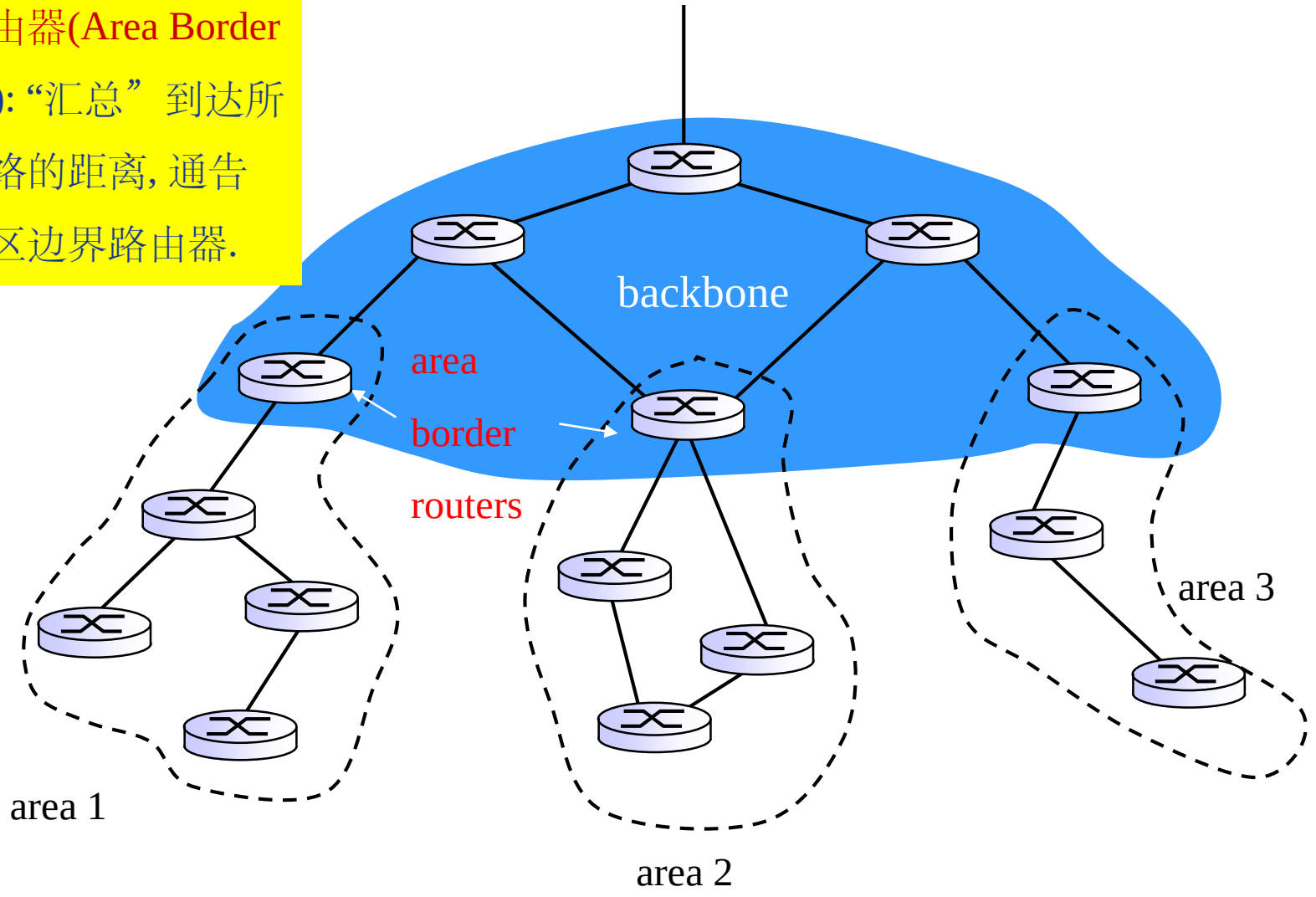
两级分层: 局部区**(Area),**
主干区**(Backbone).**

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



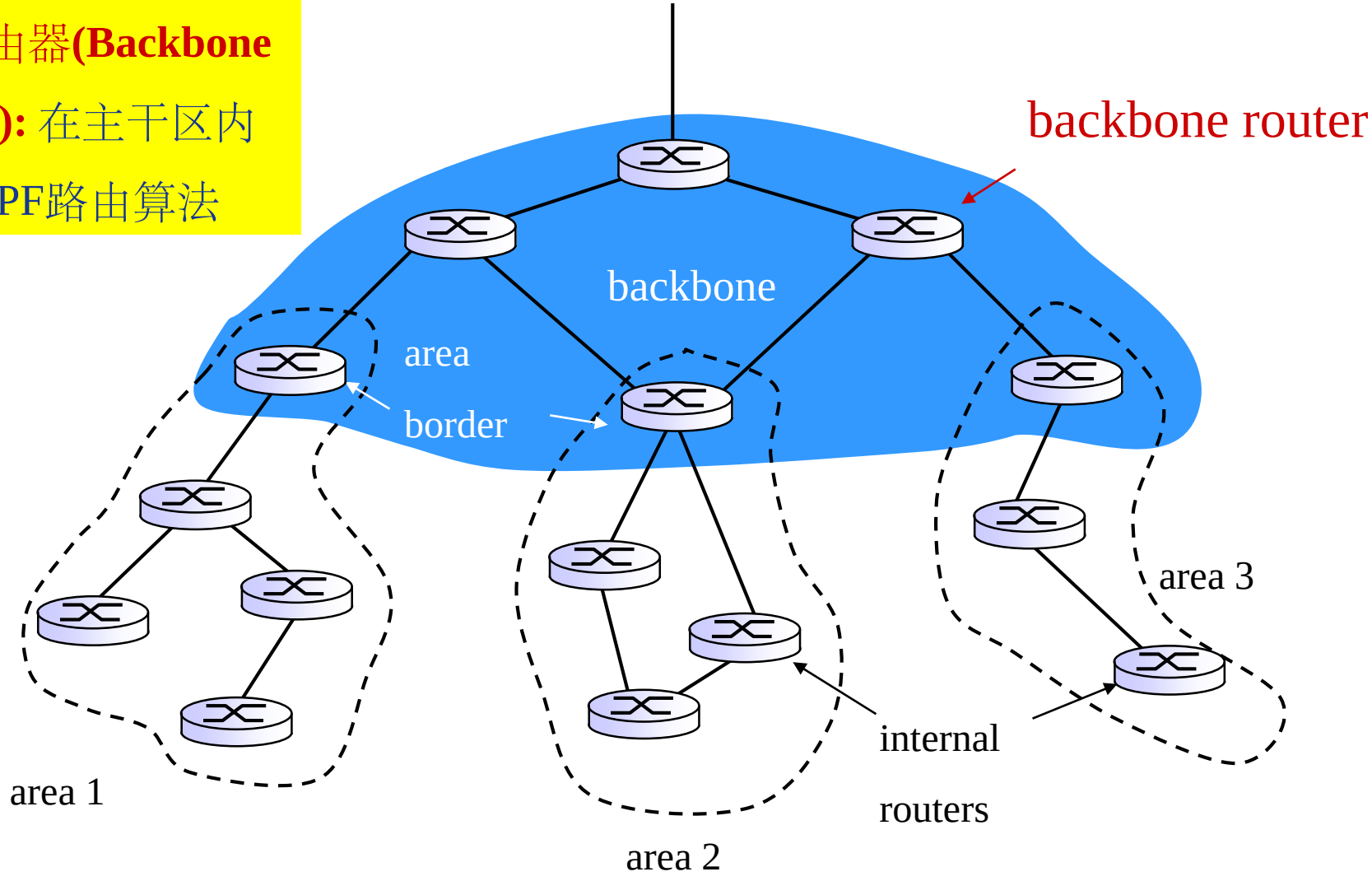
分层的OSPF

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



分层的OSPF

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

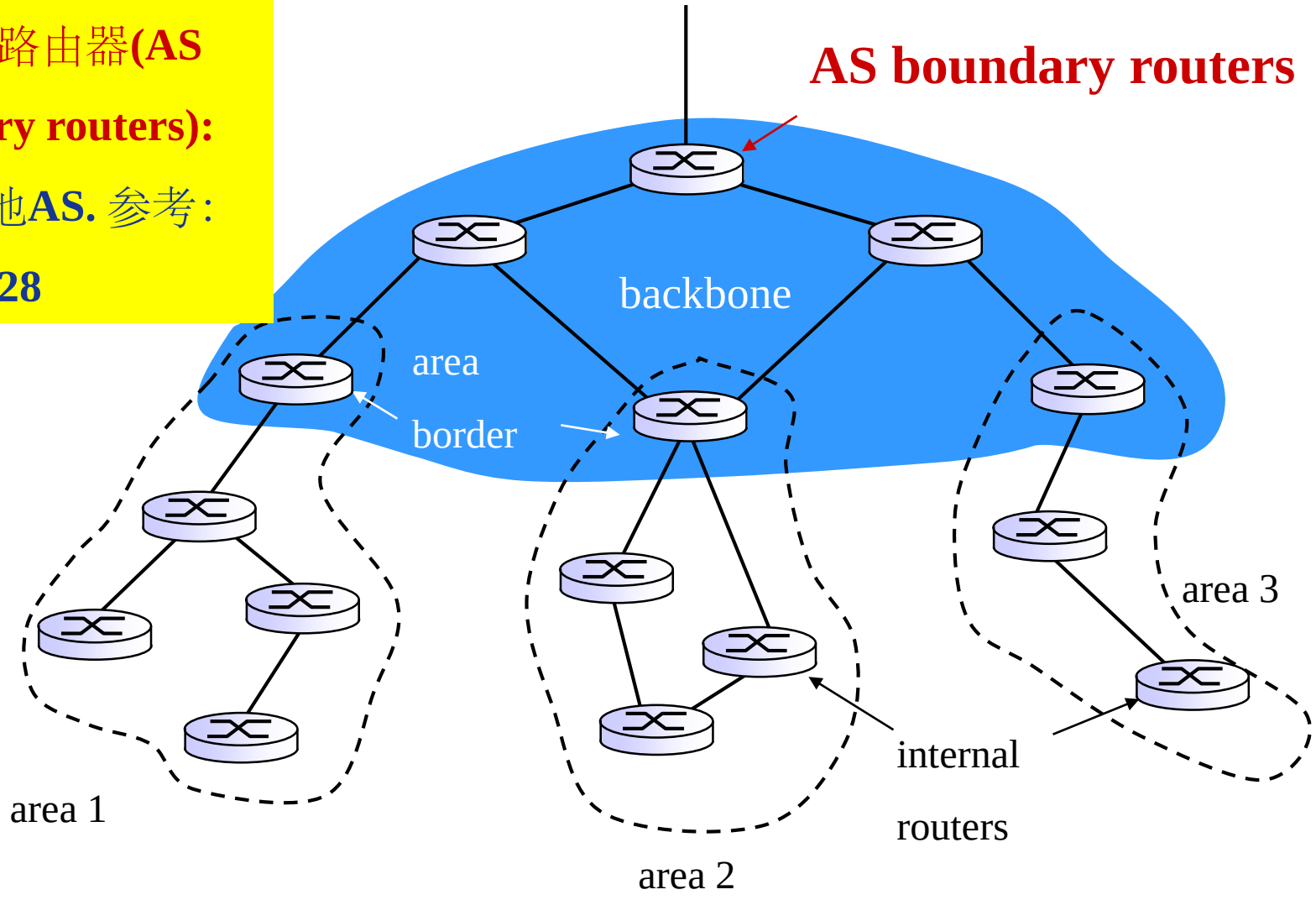


分层的OSPF

AS边界路由器(AS boundary routers):

连接其他**AS.** 参考:

RFC 2328



OSPF vs RIP 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。

- 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。

ISP之间的路由规划：BGP

- 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。
  - 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协议

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

ISP之间的路由规划：BGP

- 假设AS1内某路由器收到一个目的地址在AS1之外的数据报：

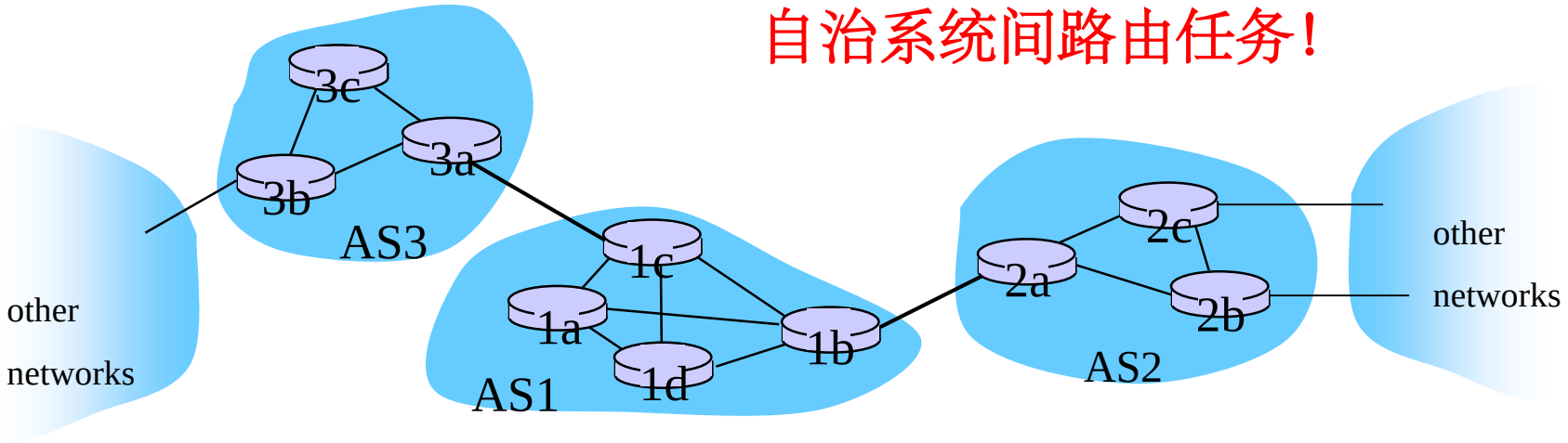
路由器应该将该数据报转发给哪个网关路由器呢？

AS1必须：

1.学习到哪些目的网络可以通过AS2到达，哪些可以通过AS3到达

2.将这些网络可达性信息传播给AS1内部路由器

自治系统间路由任务！



BGP协议

- 边界网关协议**BGP (Border Gateway Protocol):** 事实上的标准域间路由协议

•将**Internet** “粘合”为一个整体的关键

- **BGP**为每个**AS**提供了一种手段:

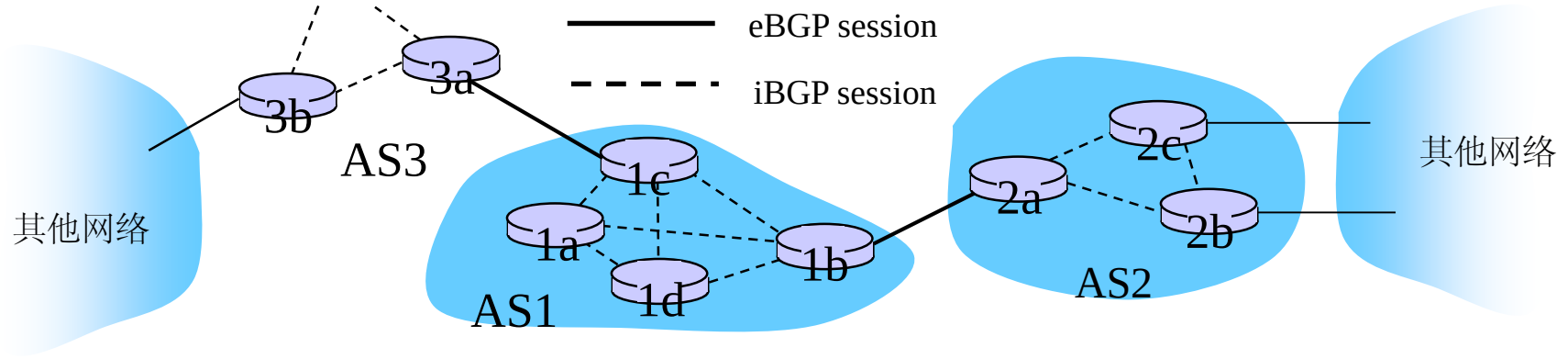
•**eBGP:** 从邻居**AS**获取子网可达性信息.

•**iBGP:** 向所有**AS**内部路由器传播子网可达性信息.

•基于可达性信息与策略，确定到达其他网络的 “好” 路径.

BGP协议

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



BGP协议

- 通告的前缀信息包括BGP属性

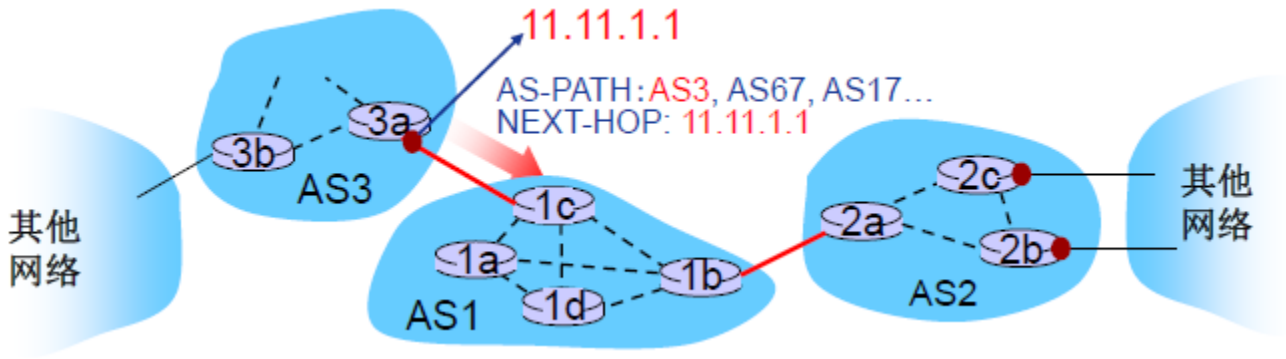
•前缀+属性= “路由”

- 两个重要属性:

•AS-PATH(AS路径): 包含前缀通告所经过的AS序列: e.g., AS 67, AS 17

•NEXT-HOP(下一跳): 开始一个AS-PATH的路由器接口，指向下一跳AS.

- 可能从当前AS到下一跳AS存在多条链路

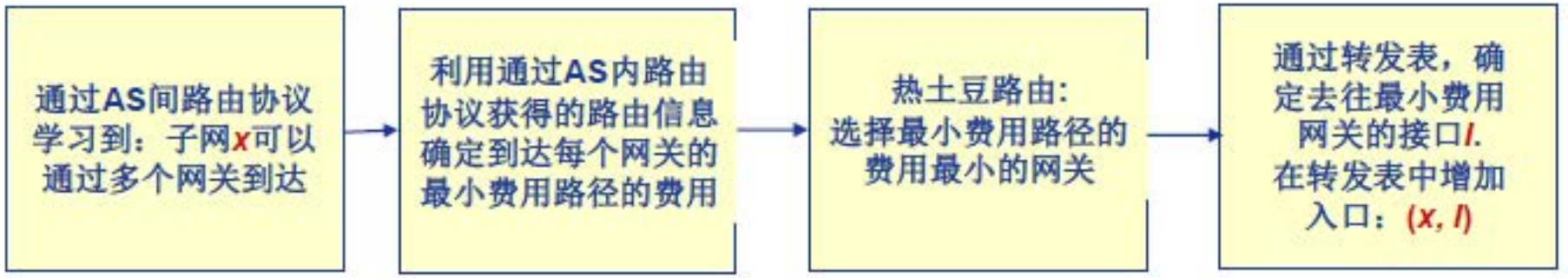


BGP路由原则：热土豆路由算法

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

•这个任务也是由AS间路由协议完成!

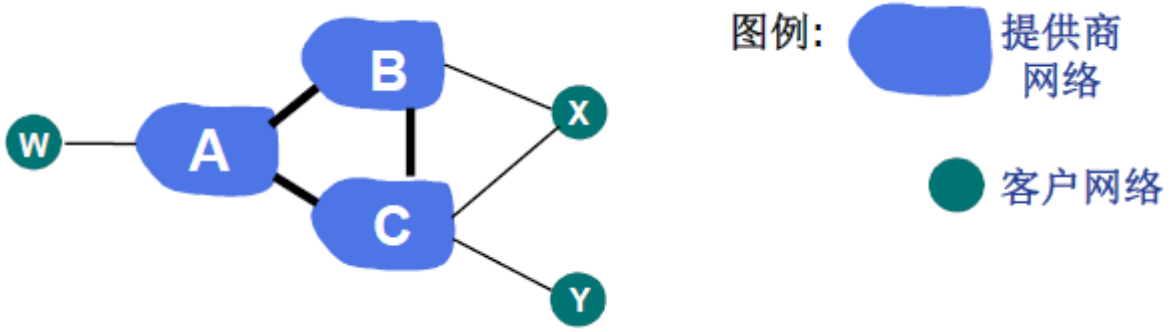
- 热土豆路由：将分组发送给最近的网关路由器.



BGP协议

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

BGP协议：路由选择策略



- A向B通告一条路径：AW
- B向X通告路径：BAW
- B是否应该向C通告路径BAW呢?

•绝不! B路由CBAW的流量没有任何“收益”，因为W和C均不是B的客户。

•B期望强制C通过A向W路由流量

•B期望只路由去往/来自其客户的流量！

BGP协议：路由选择策略

1. 路由算法：Dijkstra算法，距离向量算法；各自的优缺点
2. 路由协议：OSPF，RIP，BGP
3. 作业：p5，p7，p8，p14，p15